

# Organizing Web Information

## CS 728

Soumen Chakrabarti  
IIT Bombay

<http://www.cse.iitb.ac.in/~soumen/>

# Overview

## Overview: Main topics

- ▶ Segmenting and annotating token sequences
- ▶ Fine type tagging; entity disambiguation
- ▶ Coreference resolution; record linkage
- ▶ Closed-domain relation extraction
- ▶ Open-domain information extraction (OpenIE)
- ▶ Continuous representations of relations
- ▶ Query interpretation based on entities and types
- ▶ Tables and quantities

Also see [CourseIntroduction.ppt](#)

# High-level TOC

## Annotation:

- ▶ Identifying token segments as entity mentions
- ▶ Ditto for relations
- ▶ Annotation vs. extraction
- ▶ Closed vs. open domain

## Disambiguation:

- ▶ Supervised: against entities in a catalog
- ▶ Unsupervised: cluster mentions referring to same entity

## Search:

- ▶ Relational
- ▶ XML, trees, twigs
- ▶ Proximity in text and general graphs

## Prerequisites

- ▶ Statistics: distribution, moments, confidence
- ▶ Machine learning: features, classification, clustering
- ▶ Strings, sets, relations, graphs
- ▶ Basic knowledge of relational databases
- ▶ Basic experience with text tokenization, indexing, search

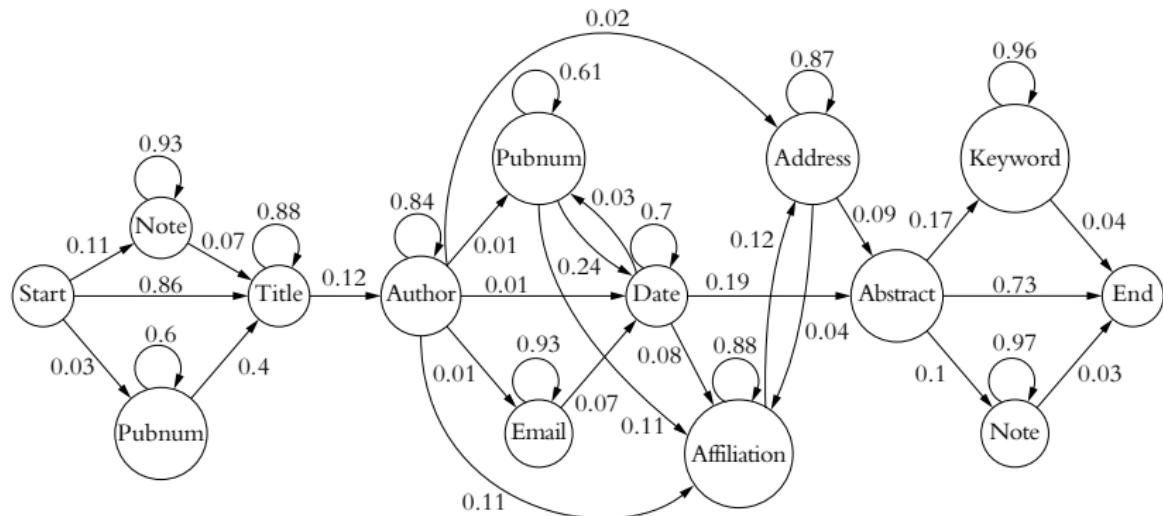
## References

## Sequence and graph labeling

## Tagging token spans

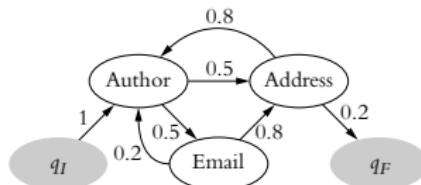
- ▶ Text modeled as a sequence of tokens
- ▶ Tokens may include punctuations, different kinds of spaces, etc.
- ▶ Restricted set of types or domains
  - ▶ Classic application in NLP: **part-of-speech (POS) tagging**, with token labels like NN, NNS, VB, VBD, VBP, PP\$, JJ, JJR, RB, etc.
  - ▶ Named entity recognition (NER): person, place, organization, date, paper title, conference venue, journal name, page range, number and street name
- ▶ *Closed domain* means examples available for each label
- ▶ A test instance is an unlabeled token sequence
- ▶ The job is to mark each token with a label seen in training instances
- ▶ In NER, most tokens may have a default “none” label

## Markov models [2]

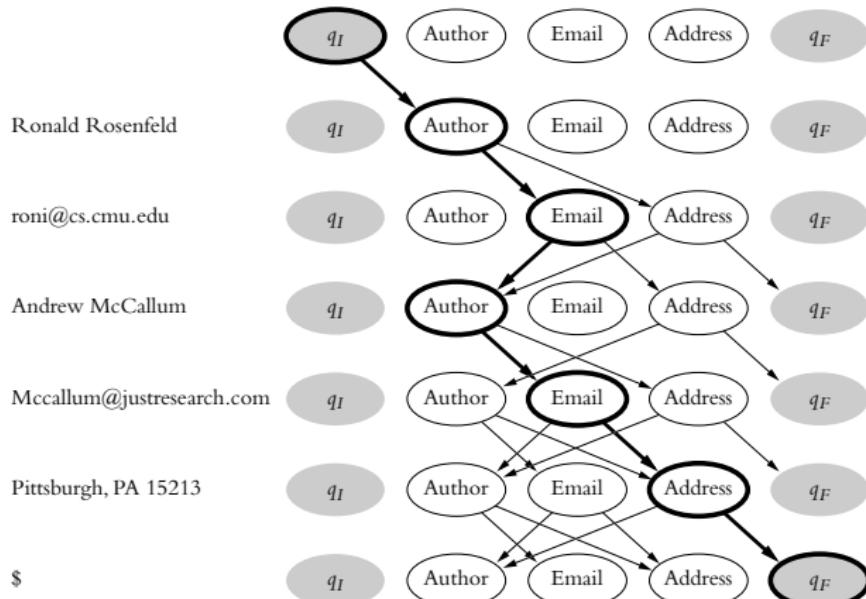


- ▶ System is in one of  $M$  states  $y_t$
- ▶ In each state, emits  $x_t$  one of  $N$  symbols
- ▶ Then moves to one of  $M$  states  $y_{t+1}$
- ▶ Many state transitions disallowed

# Viterbi decoding



x1 Ronald Rosenfeld  
x2 roni@cs.cmu.edu  
x3 Andrew McCallum  
x4 mccallum@justresearch.com  
x5 Pittsburgh, PA 15213  
x6 \$



## Dynamic programming

- ▶ Sequence of length  $T$ , pad with special positions 0 and  $T + 1$ , let  $t \in [0, T + 1]$  be a token position
- ▶ Let  $m \in [M]$  be a state; add special states  $y_0 = q_I, y_{T+1} = q_F$  at positions 0 and  $T + 1$
- ▶ Define transitions to/from special states and emissions from special states suitably
- ▶  $\psi(m, m') = \Pr(m \rightarrow m')$  is the probability of a state transition from  $m$  to  $m'$
- ▶  $\lambda(m, x) = \Pr(m \uparrow x)$  is the probability of emitting token  $x$  while in state  $m$
- ▶ Let  $V(t, m)$  be the largest probability of a state transition path ending in state  $m$  at position  $t$
- ▶ Base case  $V(0, y_0) = 1$

## Dynamic programming (2)

- ▶ For time steps  $t > 0$  and  $m' \in [M]$ :

$$V(t, m') = \max_{m \in [M]} V(t - 1, m) \psi(m, m') \lambda(m', x_t)$$

- ▶ Highest probability path can be read off as  $V(T + 1, q_F)$
- ▶ In actual code, multiplying small probabilities will underflow rapidly, so use  $\log \psi(m, m')$  and  $\log \lambda(m, x)$

$$V(0, q_I) = 0$$

$$V(0, m) = -\infty \quad \text{for } m \neq q_I$$

$$V(t, m') = \max_{m \in [M]} V(t - 1, m) + \log \psi(m, m') + \log \lambda(m', x_t)$$

- ▶ Keep track of which  $m$  maximized  $V(t, m')$ , so we can recover the path

## Max probability $\Leftrightarrow$ shortest path

- ▶ Each edge  $(u, v)$  has a probability  $p(u, v)$
  - ▶ Probability of a path is the product of edge probabilities
  - ▶ The goal is to find the **max** probability path
  - ▶ Instead, let the edge have a weight  $w(u, v) = -\log p(u, v) \geq 0$
  - ▶ Goal changes to conventional **shortest** path
  - ▶ If there are  $k$  transitions from state  $m$  to  $m'$ ,  $-\log p(m, m')$  is accumulated  $k$  times
  - ▶ Similarly for symbol emission probabilities
- ▶ HW What if you wanted to list the top 10 shortest paths?

## Estimating $\Pr(m \rightarrow m')$ and $\Pr(m \uparrow x)$

- ▶ If we had completely labeled data sets  $(\vec{x}, \vec{y})$ , then this amounts to just counting
  - ▶ The number of times symbol  $x$  was emitted while in state  $m$
  - ▶ The number of times state  $m$  transitioned to state  $m'$  and then normalizing suitably to probabilities
- ▶ Care is required with smoothing counts for rare transition and emission events — we covered this in Web Search and Mining (A)
- ▶ Given incompletely or not labeled sequences, EM is a common technique:
  - ▶ Start with a “good enough” initial estimate of these probabilities
  - ▶ Use these to (re)label given sequences, getting distributions over states
  - ▶ Use these distributions to soft-count and update probability estimates

## B-I-O and B-C-E-O state models [3]

- ▶ States none, person, place, epoch
- ▶ Roy went to Rio de Janeiro in 1994
- ▶ Change state/label definitions to placeB, placel, epochB, epochl, ..., other=none
- ▶ “Begin”, “In”, “Other/out”
- ▶ Also used: begin-continue-end, other states
- ▶ Emission distributions can be fine tuned to (three kinds of) positions inside long mentions
- ▶  $B \rightarrow I$ ,  $I \rightarrow I$ ,  $I \rightarrow O$  transition probabilities can be tuned separately
- ▶ No  $I \rightarrow B$  or  $O \rightarrow I$  transitions allowed

## Surface features

- ▶ Would like to think of  $x_t$  not as a symbol from a set of  $N$  symbols
- ▶ But as a **feature vector** with  $F$  features
- ▶ I.e. we are interested in various **properties** of token  $x_t$  rather than the identity of  $x_t$  itself
- ▶  $x_t[f]$  is the  $f$ th feature at position  $t$
- ▶ Examples of features: hasCap, isAllCap, isXxx, hasDigit, isAllDigits, isDDDD, part of speech
- ▶ Does  $x_t$  start with an uppercase letter and end with “sky”, “ski”, or “jee”?
- ▶ Note that the identity of  $x_t$  itself can still be passed on through *lexicalized* features: “is  $x_t$  equal to coffee?”
- ▶ I.e., if there are 20,000 words in the vocabulary and 50 derived features,  $F = 20050$

## Limitations of generative models: feature dependence

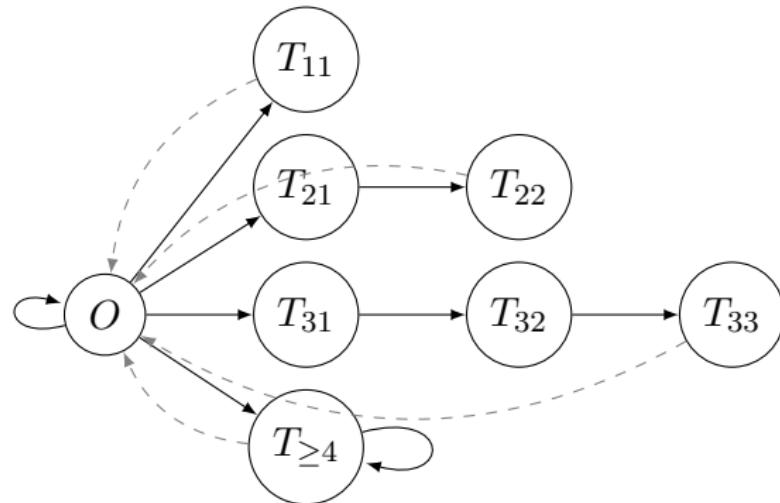
- ▶ Some of the  $F$  features are heavily correlated
  - ▶  $\text{isAllCap} \Rightarrow \text{hasCap}$
  - ▶  $\text{isDDDD} \Rightarrow \text{isAllDigits}$
  - ▶  $\text{isProperNoun}$  almost always implies  $\text{isXxx}$
- ▶ Standard generative HMM says

$$\Pr(\vec{x}, \vec{y}) = \Pr(y_0) \prod_{t=1}^T \Pr(y_t | y_{t-1}) \Pr(x_t | y_t)$$

- ▶ Now  $\Pr(x_t | y_t)$  changes from univariate multinomial to a vector of (binary, highly correlated) features
- ▶  $\prod_{t=1}^T \Pr(y_t | y_{t-1}) \left[ \prod_{f=1}^F \Pr(x_t[f] | y_t) \right]$  too crude
- ▶ Prefer to model  $\Pr(\vec{y} | \vec{x})$  rather than model  $\Pr(\vec{x} | \vec{y})$  and use Bayes rule (naive Bayes vs. logistic regression)

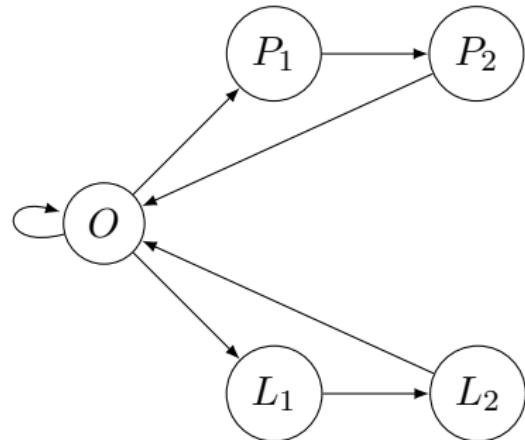
## Limitations of generative models: segment lengths

- ▶ Long named entity spans like “paper title”, “movie title”, or “organization name” modeled as self-loop on corresponding states
- ▶ Implies that the number of tokens in such spans is geometrically distributed
- ▶ Assumption does not match data
- ▶ Can patch with a collection of chains; training may suffer



## Limitations of generative models: Label bias<sup>1</sup> [4]

- ▶ Labels other ( $O$ ), person ( $P$ ), location ( $L$ )
  - ▶ Every person and location name has two tokens
  - ▶ States specialized to  $P_1, P_2; L_1, L_2$
  - ▶ **Harvey Ford**: person 9 times, location 1 time
  - ▶ **Harvey Park**: person 1 time, location 9 times
- ⇒ No clue to choose correct edge out of  $O$  after seeing **Harvey**



$$\Pr(P_1|O, \text{Harvey}) = \Pr(L_1|O, \text{Harvey}) = 1/2$$

$$\Pr(P_2|P_1, *) = \Pr(L_2|L_1, *) = 1$$

- ▶ Normalized transition probability  $\prod_t \Pr(y_t|y_{t-1})$  is culprit

<sup>1</sup>Example by Ralph Grishman

## From local to global normalization

- ▶ Recall multinomial naive Bayes text classification

$$\Pr(y|x) \propto \Pr(y) \prod_f \Pr(f|y)^{x[f]} \propto \Pr(y) \exp \left[ \sum_f x[f] \log \Pr(f|y) \right]$$

- ▶ Here  $\Pr(f|y)$  is locally normalized: for each  $y$ ,  $\sum_f \Pr(f|y) = 1$
- ▶ Logistic regression replaced this with general weights

$$\Pr(y|x) \propto \exp(x \cdot w_y) = \exp \left[ \sum_f x[f] \textcolor{red}{w_y[f]} \right] = \prod_f \exp[x[f] w_y[f]]$$

- ▶ No local normalization requirement on  $w_y$ ; only  $\sum_y \Pr(y|x)$  normalized if needed
- ▶ Will repeat this recipe for transitions and features

## Transition and features for one step

- ▶ Design a function  $\varphi : \mathcal{X} \times [M] \times [M] \rightarrow \{0, 1\}^{MF + M^2}$   
initialize  $A \leftarrow 0^{M \times F}, B \leftarrow 0^{M \times M}$   
 $B[m, m'] \leftarrow 1$   
**for** each feature  $f$  **do**  
     $A[m, f] \leftarrow x[f]$   
**return**  $[A, B]$
- ▶ Typically called as  $A_t, B_t \leftarrow \varphi(x_t, y_{t-1}, y_t)$
- ▶ Let model weights be  $w = (\alpha, \beta)$  where  $\alpha \in \mathbb{R}^{M \times F}, \beta \in \mathbb{R}^{M \times M}$
- ▶ The score for the  $t$ -th step is written as  $\exp \left[ [A_t, B_t] \otimes [\alpha, \beta] \right]$   
where  $\otimes$  is elementwise inner-product between matrices, i.e.,  
$$\exp \left[ \sum_{m,f} A_t[m, f] \alpha[m, f] + \sum_{m,m'} B_t[m, m'] \beta[m, m'] \right]$$
$$= \exp \left[ \sum_f x_t[f] \alpha[m, f] + \beta[y_t, y_{t-1}] \right] = e^{\beta[y_t, y_{t-1}]} \prod_f \exp(\cdots)$$

## Summing over steps $t$

- ▶ Because of the log-linear form, we can write

$$A = \sum_t A_t, \quad B = \sum_t B_t, \quad \text{and then}$$

$$\Pr(\vec{y}|\vec{x}) \propto \exp\left[ [A, B] \otimes [\alpha, \beta] \right] = \prod_t e^{\beta[y_t, y_{t-1}]} \prod_{t,f} \exp(\dots)$$

- ▶  $A[m, f]$  is the number of times feature  $f$  was “fired” while in state  $m$
- ▶  $B[m, m']$  is the number of transitions from  $m'$  to  $m$
- ▶ To ease notation, let

$$[A, B] = \phi(\vec{x}, \vec{y}) = \sum_t \varphi(x_t, y_{t-1}, y_t)$$

$$w = [\alpha, \beta]$$

## Summing over steps $t$ (2)

- ▶ Summarizing ...
- ▶  $\vec{x} = (x_t : t = 1, \dots, T)$  is the sequence of visible symbols
- ▶ Each position is associated with  $F$  binary (say) features
- ▶  $\vec{y} = (y_t : t = 1, \dots, T)$  is the label/state sequence
- ▶ Suppose there are  $M$  states;  $m \in [1, M]$
- ▶ Define a **feature map**  $\phi(x, y) \in \mathbb{R}^d$
- ▶ From labeled training data  $x^\ell, y^\ell : \ell = 1, \dots, L$ , learn a **model**  $w \in \mathbb{R}^d$
- ▶ Given test sequence  $\vec{x}$ , predict label sequence  
$$\arg \max_{\vec{y}} w^\top \phi(\vec{x}, \vec{y})$$
- ▶ Enumerating all  $\vec{y}$  not practical, but for chain (and some other) dependency graphs, dynamic programming suffices

## Inspecting more/all of $\vec{x}$

- ▶ Can generalize to  $\varphi(\textcolor{red}{t}, \vec{x}, m', m)$  to look at more or all of  $\vec{x}$ , provided we retain  $m' = y_{t-1}$  and  $m = y_t$
- ▶ Can also change the behavior of  $\varphi$  with  $t$ , e.g., make some states, transitions or emissions more or less likely near specific times/offsets  $t$
- ▶ Another application in the next slide
- ▶ This sort of enhancement usually needs massive state-space blowup in conventional HMMs

## Features between $x_{t\pm 1}$ and $y_t$

- ▶ So far we have limited interaction between  $\vec{x}$  and  $\vec{y}$  to individual positions, e.g.,  $x_t$  and  $y_t$
- ▶ States none, person, place, epoch
  - ▶ Roy went to Rio de Janeiro in 1994
- ▶ “at” or “to” often precede place
- ▶ “on” or “in” often precede epoch
- ▶ But the state of at, to, on, in are all ‘none’
- ▶ Define features between  $x_{t\pm 1}$  and  $y_t$ :
  - ▶  $x_{t-1} = \text{"at"} \wedge y_t = \text{placeB}$
  - ▶  $x_{t+1} = \text{"said"} \wedge y_t = \text{person!}$
- ▶ Effectively multiplies  $F$  by some (small) factor

## Inference

- ▶ Given a partial label sequence  $(y_1, \dots, y_{t-1}, y_t = m)$  of length  $t$
- ▶ Let  $v(t, m)$  be the maximum value of  $w^\top \phi(\vec{x}, (y_1, \dots, y_{t-1}, y_t = m))$  over all possible states  $y_1, \dots, y_{t-1}$  and  $y_t$  pinned to  $m$
- ▶ Base case  $v(0, m) = 0$  for all  $m$
- ▶ Recurrence step for  $t > 0$ :

$$v(t, m) = \max_{m'} v(t - 1, m') + w \cdot \varphi(t, x_t, m', m)$$

- ▶ Final solution is (the path traced to get from  $y_0$  at time 0 to)  $\max_m v(T, m)$
- ▶ Viterbi's dynamic programming algorithm

## Discriminative training of $w$ [5]

- ▶ For each sequence  $\vec{x}^\ell$ , there is (say) one correct labeling  $\vec{y}^\ell$ ; **all other** (exponentially many) labelings  $\vec{y} \neq \vec{y}^\ell$  are incorrect
- ▶ Want to fit  $w$  such that

$$\forall \ell, \forall \vec{y} \neq \vec{y}^\ell : w^\top \phi(\vec{x}^\ell, \vec{y}^\ell) > w^\top \phi(\vec{x}^\ell, \vec{y})$$

- ▶ The worse  $\vec{y}$  is, compared to  $\vec{y}^\ell$ , the bigger we want the gap to be
- ▶ **Loss function**  $\Delta(\vec{y}^\ell, \vec{y}) \geq 0$ ;  $\Delta(\vec{y}, \vec{y}) = 0$
- ▶ E.g., Hamming loss

$$\forall \ell, \forall \vec{y} \neq \vec{y}^\ell : w^\top \phi(\vec{x}^\ell, \vec{y}^\ell) \geq w^\top \phi(\vec{x}^\ell, \vec{y}) + \Delta(\vec{y}^\ell, \vec{y})$$

- ▶ Allow a **slack**  $\xi_\ell \geq 0$ :

$$\forall \ell, \forall \vec{y} \neq \vec{y}^\ell : w^\top \phi(\vec{x}^\ell, \vec{y}^\ell) + \xi_\ell \geq w^\top \phi(\vec{x}^\ell, \vec{y}) + \Delta(\vec{y}^\ell, \vec{y})$$

## Discriminative training of $w$ [5] (2)

- ▶ Objective has two parts:
  - ▶  $\sum_\ell \xi_\ell$  representing upper bound on training loss
  - ▶  $\frac{1}{2} \|w\|_2^2$ , the model complexity

usually balanced with a tuned magic constant  $C$

$$\min_{\xi \geq \vec{0}; w} \frac{1}{2} \|w\|_2^2 + C \sum_\ell \xi_\ell \quad \text{s.t.}$$

$$\forall \ell, \forall \vec{y} \neq \vec{y}^\ell : w^\top \boxed{\phi(\vec{x}^\ell, \vec{y}^\ell) - \phi(\vec{x}^\ell, \vec{y})} \geq \Delta(\vec{y}^\ell, \vec{y}) - \xi_\ell$$

- ▶  $M^T - 1$  constraints per training sequence!
- ▶ STRUCTSVM to the rescue

## (Loss augmented) inference

- ▶ After  $w$  is trained, the **inference** problem is to find  $\arg \max_{\vec{y} \in \mathcal{Y}} w^\top \phi(\vec{x}, \vec{y})$  for test  $x$
- ▶  $M$  states, sequence length  $T \implies |\mathcal{Y}| = M^T$
- ▶ Viterbi dynamic programming takes  $O(M^2 T)$  time
- ▶ **Loss augmented inference** is to find  $\tilde{y} = \operatorname{argmax}_{\vec{y}} w^\top \phi(\vec{x}^\ell, \vec{y}) + \Delta(\vec{y}^\ell, \vec{y})$  for training instance  $(\vec{x}^\ell, \vec{y}^\ell)$
- ▶ (I.e., find **bad**  $\tilde{y}$  whose loss and score are both large)
- ▶ If  $w \cdot \phi(\vec{x}^\ell, \tilde{y}) + \Delta(\vec{y}^\ell, \tilde{y}) < w \cdot \phi(\vec{x}^\ell, \vec{y}^\ell)$ , no bad  $\tilde{y}$  really exists, can terminate
- ▶ Otherwise, make a new constraint using  $\tilde{y}$  and optimize for  $w$  again

## Decomposable loss examples

- ▶ Can also be solved via dynamic programming for decomposable (and some non-decomposable)  $\Delta$
- ▶ All-or-nothing  $\Delta_{0/1}(\vec{y}, \vec{y}') = \llbracket \vec{y} \neq \vec{y}' \rrbracket$
- ▶ Hamming  $\Delta_h(\vec{y}, \vec{y}') = \sum_t \llbracket y_t \neq y'_t \rrbracket$
- ▶ At most three errors  $\Delta_{\leq 3}(y, y') = \llbracket \Delta_h(y, y') > 3 \rrbracket$
- ▶ Dimishing marginal annoyance:
  - ▶ In a sequence of length  $T$  there can be  $k \in [0, T]$  mistakes
  - ▶ Let the loss be  $\Gamma(\sum_t \llbracket y_t \neq y'_t \rrbracket)$
  - ▶ E.g.  $\Gamma(\bullet) = \sqrt{\bullet}$  or  $\log(1 + \bullet)$
- ▶  $V(t, m, k)$  is the largest (loss augmented) score of ending in state  $m$  at time  $t$  with  $k$  mistakes
- ▶  $V(0, y_0, 0) = 0$ ,  $V(0, \neq y_0, *) = -\infty$
- ▶  $V(t, *, > t) = -\infty$  for all  $t$  (cannot make  $> t$  mistake up to time  $t$ )

## Decomposable loss examples (2)

- ▶ Let  $\bar{y}^*$  be the gold sequence
- ▶ Suppose  $y_t^* = m$ , i.e., no mistake at time  $t$

$$V(t, m, k) = \max_{m'} \{ V(t - 1, m', k) + w \cdot \varphi(x_t, m', m) \}$$

- ▶ Suppose  $y_t^* \neq m$ , i.e., mistake at time  $t$

$$\begin{aligned} V(t, m, k) = \max_{m'} & \left\{ V(t - 1, m', \textcolor{red}{k - 1}) + w \cdot \varphi(x_t, m', m) \right. \\ & \left. + \Gamma(k) - \Gamma(k - 1) \right\} \end{aligned}$$

- ▶ Because loss is nonlinear, must remember and take off the previous loss  $\Gamma(k - 1)$  and add current accumulated loss  $\Gamma(k)$

▶ HW Check, complete and implement

- ▶ Useful for robust learning with some fraction of ‘hopeless’ training sequences that just add a noise floor to training loss

## Conditional probability model

- ▶ Another option is to model a conditional probability:  
 $\Pr(\vec{y}|\vec{x}) \propto \exp(w^\top \phi(\vec{x}, \vec{y}))$
- ▶ Sometimes written as  $\Pr(\vec{y}|\vec{x}; w)$  to make model  $w$  explicit
- ▶ Inference is  $\arg \max_{\vec{y}} \Pr(\vec{y}|\vec{x}) = \arg \max_{\vec{y}} w^\top \phi(\vec{x}, \vec{y})$  which is exactly the same as in max-margin training
- ▶ Using a normalizing factor  $Z_w(\vec{x}) = \sum_{\vec{y}} \exp(w^\top \phi(\vec{x}, \vec{y}))$ , we can write  $\Pr(\vec{y}|\vec{x}) = \exp(w^\top \phi(\vec{x}, \vec{y}))/Z_w(\vec{x})$
- ▶ Training  $w$  amounts to optimizing

$$\begin{aligned}\arg \max_w \prod_{\ell} \Pr(\vec{y}^\ell | \vec{x}^\ell) &= \arg \max_w \sum_{\ell} \log \Pr(\vec{y}^\ell | \vec{x}^\ell) \\ &= \arg \max_w \sum_{\ell} w^\top \phi(\vec{x}^\ell, \vec{y}^\ell) - \log Z_w(\vec{x}^\ell)\end{aligned}$$

- ▶ Concave, global maximum, use Newton method

## Computing $\nabla_w \log Z_w(x)$

- ▶ First let us find  $Z_w(\vec{x})$  in polynomial time
- ▶ Recall  $\phi(\vec{x}, \vec{y}) = \sum_{1 \leq t \leq T} \varphi(x_t, y_{t-1}, y_t)$
- ▶ Define partial sums

$$\phi_{[1:t]}(\vec{x}[1:t], \vec{y}[1:t]) = \sum_{1 \leq \tau \leq t} \varphi(x_\tau, y_{\tau-1}, y_\tau) \quad \text{and}$$

$$\alpha(t, m) = \sum_{\gamma_1, \dots, \gamma_{t-1}} \exp(w \cdot \phi_{[1:t]}(\vec{x}[1:t], (\gamma_1, \dots, \gamma_{t-1}, m)))$$

- ▶ By definition,  $Z_w(\vec{x}) = \sum_m \alpha(T, m)$
- ▶ Base case  $\alpha(0, m) = 1$  for all  $m$
- ▶ Recurrence (last transition  $m' \rightarrow m$ ):

$$\alpha(t, m) = \sum_{m'} \alpha(t-1, m') \exp(w \cdot \varphi(x_t, m', m))$$

## Computing $\nabla_w \log Z_w(x)$ (2)

- ▶  $\nabla_w \log Z_w(x) = \frac{1}{Z_w(x)} \nabla_w Z_w(x) =$   
 $\frac{1}{Z_w(x)} \sum_{\vec{y}} \nabla_w \exp(w \cdot \phi(\vec{x}, \vec{y})) = \frac{1}{Z_w(x)} \sum_{\vec{y}} \phi(\vec{x}, \vec{y}) \exp(w \cdot \phi(\vec{x}, \vec{y}))$
- ▶ Btw,  
 $\frac{1}{Z_w(x)} \sum_{\vec{y}} \phi(\vec{x}, \vec{y}) \exp(w \cdot \phi(\vec{x}, \vec{y})) = \sum_{\vec{y}} \phi(\vec{x}, \vec{y}) \frac{\exp(w \cdot \phi(\vec{x}, \vec{y}))}{Z_w(x)} =$   
 $\sum_{\vec{y}} \phi(\vec{x}, \vec{y}) \Pr(\vec{y} | \vec{x}; w) = \mathbb{E}_{\Pr(\vec{y} | \vec{x}; w)} \phi(\vec{x}, \vec{y})$
- ▶ Note that this is an expected feature vector
- ▶ We have already computed  $Z_w(\vec{x})$ , now we need to compute  $\sum_{\vec{y}} \phi(\vec{x}, \vec{y}) \exp(w \cdot \phi(\vec{x}, \vec{y}))$  efficiently
- ▶ Again define a partial sum over positions

$$\begin{aligned}\eta(t, m) &= \sum_{\gamma_1, \dots, \gamma_{t-1}} \phi_{[1:t]}(\vec{x}[1:t], (\gamma_1, \dots, \gamma_{t-1}, m)) \\ &\quad \exp(w \cdot \phi_{[1:t]}(\vec{x}[1:t], (\gamma_1, \dots, \gamma_{t-1}, m)))\end{aligned}$$

- ▶ By definition, our goal is  $\sum_m \eta(T, m)$
- ▶ Base case  $\eta(0, m) = \vec{0}$  for all  $m$

## Computing $\nabla_w \log Z_w(x)$ (3)

- ▶ By unrolling out once,  $\eta(t, m) =$

$$\sum_{m'} \sum_{\gamma[1:t-2]} \left( \phi_{[1:t-1]}(\vec{x}[1:t-1], \underline{\gamma_1}, \dots, \underline{\gamma_{t-2}}, m') + \varphi(x_t, m', m) \right) \times \exp [w \cdot (\text{same})]$$

- ▶ HW Simplifying, we get the recurrence

$$\eta(t, m) = \sum_{m'} \left[ \eta(t-1, m') + \alpha(t-1, m') \varphi(t, x_t, m', m) \right] e^{w \cdot \varphi(x_t, m', m)}$$

- ▶ Now we can do gradient descent, e.g., AdaGrad
- ▶ Given global convexity, can take advantage of second order methods like L-BFGS

## Modeling long-range influence

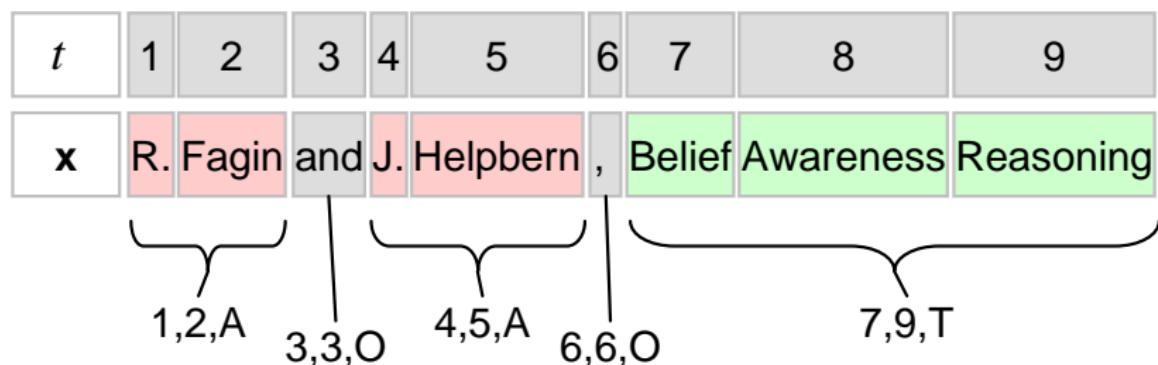
- ▶ Long segments — can still do dynamic programming, at increased cost
- ▶ Can extend from linear chains to trees without pain — (probabilistic) context free grammars
- ▶ Same token distributed through a document should get correlated labels — general graph, intractable, can only approximate

## Segment CRF motivation

- ▶ Suppose the “paper title” state has a self-loop with probability 0.9
- ▶ Then the number of tokens in a paper title follows a geometric distribution with mean 10
- ▶ Real data may not support a geometric distribution
- ▶ Many long text fields like book or movie title comprise ordinary words (e.g., Gone with the Wind)
- ▶ Often easier to recognize because of close phrase match with a dictionary
- ▶ Or anomalous words just before and after:  
... the script **for** [GWTW], **he** had no idea ...  
for Gone and wind, **he**
- ▶ Can't model this at a single token level

## Segment CRF model [7]

- ▶ A **segment**  $s_j$  is defined by start token offset  $\ell_j$  and end token offset  $u_j$
- ▶  $y_t$  replaced with  $s_j = \langle \ell_j, u_j, y_j \rangle$
- ▶ Feature function extended to  $\varphi(\ell, u, \vec{x}, m', m)$
- ▶ Usually instantiated to  $\varphi(\ell_j, u_j, x_{\ell_j} \dots x_{u_j}, y_{j-1}, y_j)$
- ▶ Vector of segments called  $\vec{s}$  similar to  $\vec{y}$



## Inference for segment CRF

- ▶ Suppose  $J$  is the length of the longest allowed segment
- ▶ Let  $v(t, m)$  be the best score of segmentations ending at  $t$  with label  $m$
- ▶  $v(0, m) = 0$  for all  $m$
- ▶ If  $t < 0$  then  $v(t, m) = -\infty$
- ▶ In general for  $t > 0$ ,  $v(t, m) =$

$$\max_{m'} \max_{t'=t-J}^{t-1} v(t', m') + w \cdot \varphi(t' + 1, t, \vec{x}, m', m)$$

## Limitations of linear segmentation [8]

- ▶ BizContact → BizName Address BizPhone
- ▶ PersonalContact → PersonName Address HomePhone
- ▶ If we see a 1-800 number, the name is more likely to be a business name
- ▶ Conversely, “Associates” in name makes the phone number more likely to be a business number
- ▶ Business numbers are more likely to have “1-800” in them

Fred Jones

10 Main St.

Cambridge, MA 02146

(425) 994-8021

Fred Jones and Associates

10 Main St.

Cambridge, MA 02146

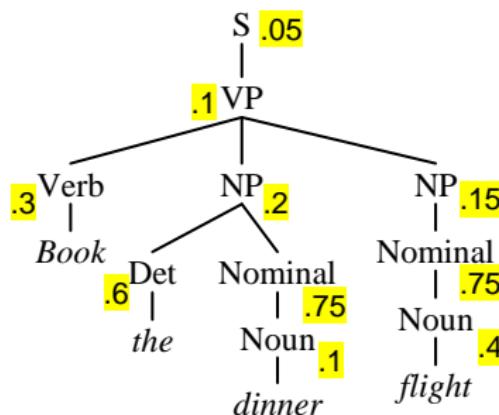
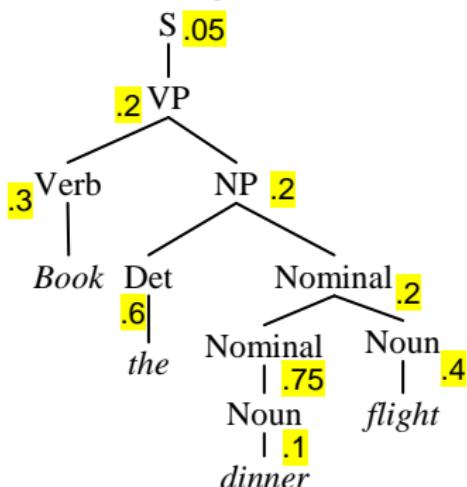
1-800-555-1212

## Probabilistic context free grammar (PCFG)

- ▶ A context-free grammar (CFG) is a 4-tuple  $G = (N, \Sigma, R, S)$  where:
  - ▶  $N$  is a finite set of non-terminal symbols.
  - ▶  $\Sigma$  is a finite set of terminal symbols.
  - ▶  $R$  is a finite set of rules of the form  $X \rightarrow Y_1 Y_2 \dots Y_n$ , where  $X \in N$ ,  $n \geq 0$ , and  $Y_i \in N \cup \Sigma$  for  $i = 1, \dots, n$ . For simplicity we will assume that  $n = 1$  and  $n = 2$  for productions to terminals and non-terminals respectively.
  - ▶  $S \in N$  is a distinguished start symbol.
- ▶ A PCFG, in addition, has a parameter  $q(\alpha \rightarrow \beta) \geq 0$  for each rule  $\alpha \rightarrow \beta \in R$
- ▶ Interpreted as the conditional probability of choosing this rule in a left-most derivation, given that the non-terminal being expanded is  $\alpha$
- ▶ For any  $X \in N$ ,  $\sum_{\beta} q(X \rightarrow \beta) = 1$

# PCFG example

Rules	$q$
$S \rightarrow VP$	.05
$VP \rightarrow Verb\ NP$	.2
$VP \rightarrow Verb\ NP\ NP$	.1
$NP \rightarrow Nominal$	.15
$NP \rightarrow Det\ Nominal$	.2
$Nominal \rightarrow Nominal\ Noun$	.2
$Nominal \rightarrow Noun$	.75
<hr/>	
$Verb \rightarrow book$	.3
$Det \rightarrow the$	.6
$Noun \rightarrow dinner$	.1
$Noun \rightarrow flight$	.4



## PCFG inference

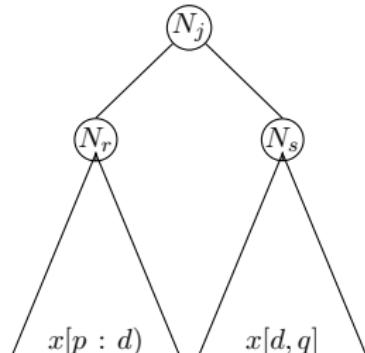
- ▶ What is the probability of a sentence given a PCFG, over all possible parse trees?
- ▶ What is the most likely parse tree for a sentence?
- ▶ For simplicity consider the restricted grammar

$$N_i \rightarrow w_j, \quad N_i \rightarrow w_j N_k$$

- ▶ Let input sequence be  $x_1, \dots, \underbrace{x_p, \dots, x_q}_{\text{inside}}, \dots, x_T$
- ▶ Suppose inside span was produced from nonterminal  $N_j$
- ▶ Inside probability is  $\beta_j(p, q) = \Pr(x_{[p:q]} | N_j)$

## CYK<sup>2</sup> dynamic programming algorithm

- ▶ Base case  $\beta_j(k, k) = \Pr(x_k | N_j) = \Pr(N_j \rightarrow x_k)$
- ▶ General case  $\beta_j(p, q)$  can be decomposed as
  - ▶ Pick two nonterminals to produce  $N_j \rightarrow N_r N_s$
  - ▶ Pick a split point  $[p, d)$  and  $[d, q]$
  - ▶  $N_r$  (eventually) produces  $x_{[p, d)}$
  - ▶  $N_s$  (eventually) produces  $x_{[d, q]}$



$$\beta_j(p, q) = \sum_{r,s} \sum_{p \leq d \leq q} \Pr(N_j \rightarrow N_r N_s) \beta_r(p, d) \beta_s(d + 1, q)$$

- ▶ Probability of the entire sentence is  $\beta_{\text{root}}(1, T)$
- ▶ Time  $T^3 R$  where  $R$  is the number of rules in grammar
- ▶ For single best parse, replace sums above with max
- ▶ Rule probabilities  $\Pr(N \rightarrow \zeta)$  estimated from fully labeled data as in HMMs

---

<sup>2</sup>Cocke-Kasami-Younger

# The chart

Book      the      flight      through      Houston

Each nonterminal that can produce <b>Book</b> , with probability				
	Each nonterminal that can produce <b>the</b> , with probability		Each nonterminal that can produce <b>the flight through</b> , with max probability	

## Discriminative CFGs

- ▶ Replace the probabilistic form of  $\beta_j(p, q)$  with a **score** computed from a feature vector and a model weight vector
- ▶ Let  $R_k$  be the rule  $N_i \rightarrow \zeta_j$ , where  $\zeta_j$  is a sequence of terminals and nonterminals
- ▶ Each rule  $R_k$  has associated model weight vector  $\lambda_k \in \mathbb{R}^F$ , say
- ▶ From  $\vec{x}$ , span  $[p, q]$ , and rule  $R_k$ , we extract feature vector  $\phi_k(\vec{x}, p, q) \in \mathbb{R}^F$  as well

$$S(N_i \rightarrow \zeta_j, p, q) = \lambda_k \cdot \phi_k(\zeta_j, \vec{x}, p, q)$$

- ▶ E.g., the terminals of  $\zeta_j$  will be compared to parts of  $\vec{x}$  to fire some features
- ▶ There may be features for each  $(i, r)$  pair where nonterminal  $N_r$  occurs in  $\zeta_j$
- ▶ As in conditional Markov sequence models, it is ok for  $\phi$  to look at  $\vec{x}$  outside the  $[p, q]$  span

## Inference and training

- ▶ MAP inference is as in generative PCFGs
- ▶ Here “ $\vec{y}$ ” is replaced by a parse tree expressed in a suitable structured label format
- ▶ For discriminative training, must assert constraints corresponding to “all wrong parses” — difficult
- ▶ Collin’s averaged perceptron:

**for** each instance  $(\vec{x}, y^*)$  **do**

    let  $\hat{y}$  be the best parse of  $\vec{x}$  using  $\Lambda = \{\lambda_k\}$

**if**  $\hat{y} \neq y^*$  **then**

**for** each rule  $R_k$  used in  $\hat{y}$  but not  $y^*$  **do**

**if** feature  $f$  is active/fired in  $\vec{x}$  **then**

$$\lambda_k[f] \leftarrow \lambda_k[f] - 1$$

**for** each rule  $R_k$  used in  $y^*$  but not  $\hat{y}$  **do**

**if** feature  $f$  is active/fired in  $\vec{x}$  **then**

$$\lambda_k[f] \leftarrow \lambda_k[f] + 1$$

# Dependency parsing

- ▶ PCFGs are used for **constituency parsing**, which reveal the hierarchical phrase/clause structure of the sentence; intermediate levels of the hierarchy are represented by nonterminal nodes
- ▶ In contrast, a **dependency parse** connects pairs of words in a tree structure without introducing new nodes; edges connecting words are labeled with the nature of relationship between them
- ▶ Important techniques:
  - ▶ Maximum rooted directed arborescence
  - ▶ “Deterministic” shift-reduce style
- ▶ Apart from standard NLP tasks, used for:
  - ▶ Composing continuous representation of sentences
  - ▶ Translating questions into structured query plans
  - ▶ Features in fine type tagging, relation extraction

<http://nlp.stanford.edu:8080/corenlp/>

<https://stp.lingfil.uu.se/~nivre/docs/ACLslides.pdf>

## Factoring joint distributions

- ▶ Random variables  $Y_1, \dots, Y_6 \in \{0, 1\}$  say
- ▶  $y_i$  is a specific value for variable  $Y_i$
- ▶ Joint distribution  $\Pr(Y_1 = y_1, \dots, Y_6 = y_6)$
- ▶ Suppose we can write the joint distribution in the chain form  $f_{1,2}(y_1, y_2)f_{2,3}(y_2, y_3)f_{3,4}(y_3, y_4)f_{4,5}(y_4, y_5)f_{5,6}(y_5, y_6)/Z$  where each  $f_{i,i+1} : \{0, 1\}^2 \rightarrow \mathbb{R}_+$  and  $Z$  is a normalizer
- ▶ The marginal distribution of one variable, say  $Y_1$ , is given by  $\Pr(Y_1 = y_1) = (1/Z) \sum_{y_2, y_3, y_4, y_5, y_6} \prod_{i=1}^5 f_{i,i+1}(y_i, y_{i+1})$
- ▶ As written, it seems like  $2^5$  terms in the sum have to be evaluated explicitly
- ▶ However, notice that  $\sum_{y_2, y_3, y_4, y_5, y_6} \prod_{i=1}^5 f_{i,i+1}(y_i, y_{i+1}) = \sum_{y_2, y_3, y_4, y_5} \prod_{i=1}^4 f_{i,i+1}(y_i, y_{i+1}) [\sum_{y_6} f_{5,6}(y_5, y_6)] = \sum_{y_2, y_3, y_4} \prod_{i=1}^3 f_{i,i+1}(y_i, y_{i+1}) \left[ \sum_{y_5} f_{4,5}(y_4, y_5) [\sum_{y_6} f_{5,6}(y_5, y_6)] \right]$

## Factoring joint distributions (2)

- We can compute this as follows:

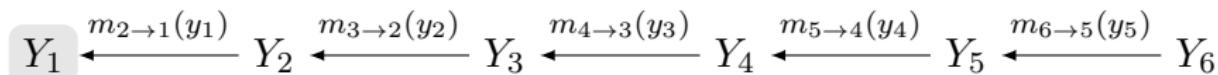
$$m_{6 \rightarrow 5}(y_5) = \sum_{y_6} f_{5,6}(y_5, y_6)$$

$$m_{5 \rightarrow 4}(y_4) = \sum_{y_5} f_{4,5}(y_4, y_5) m_{6 \rightarrow 5}(y_5)$$

$$\dots = \dots$$

$$m_{2 \rightarrow 1}(y_1) = \sum_{y_2} f_{1,2}(y_1, y_2) m_{3 \rightarrow 2}(y_2)$$

$$\Pr(Y_1 = y_1) = (1/Z) m_{2 \rightarrow 1}(y_1) = m_{2 \rightarrow 1}(y_1) / \sum_{\bullet} m_{2 \rightarrow 1}(\bullet)$$

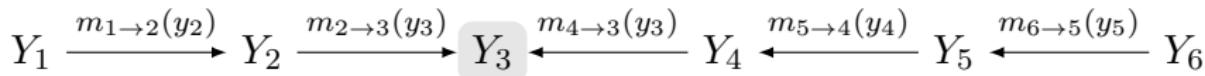


- Assuming in general  $y_i \in \mathcal{Y}_i$ , the time taken to compute  $m_{(i+1) \rightarrow i}(y_i)$  for all values of  $y_i$  is  $|\mathcal{Y}_{i+1}| |\mathcal{Y}_i|$
- Total time is like  $\sum_{i=1}^5 |\mathcal{Y}_{i+1}| |\mathcal{Y}_i|$ , which can be exponentially smaller than  $\prod_i |\mathcal{Y}_i|$  (for “larger values of 5”)

## Factoring joint distributions (3)

- ▶ Slightly different: marginal distribution of  $Y_3$  is given by

$$\Pr(Y_3 = y_3) = (1/Z) \sum_{y_1, y_2, y_4, y_5, y_6} \prod_{i=1}^5 f_{i, i+1}(y_i, y_{i+1})$$



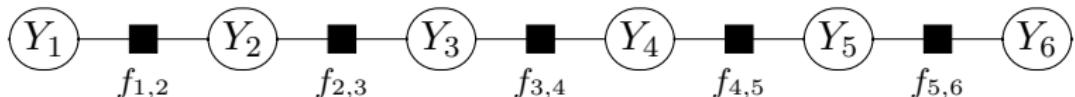
- ▶ This time, two **messages** reach  $Y_3$ :  $m_{2 \rightarrow 3}(y_3)$  and  $m_{4 \rightarrow 3}(y_3)$
- ▶ Should simply multiply them together and normalize:

$$\Pr(Y_3 = y_3) = \frac{m_{2 \rightarrow 3}(y_3) m_{4 \rightarrow 3}(y_3)}{\sum_{\bullet \in \mathcal{Y}_3} m_{2 \rightarrow 3}(\bullet) m_{4 \rightarrow 3}(\bullet)}$$

- ▶ Extends to trees
  - ▶ Given  $Y_{\text{root}}$  whose marginal distribution we want
  - ▶ Conduct DFS rooted at  $Y_{\text{root}}$
  - ▶ Start fanning in messages in reverse DFS numbering order

## Factor graph [9]

- ▶ Chain graph in standard notation:



- ▶ In general, bipartite graph, factor layer  $F$ , variable layer  $V$
- ▶ Assume all variables unobserved (will generalize later)
- ▶ Factors denoted  $a, b, \dots$ , variables  $u, v, \dots$
- ▶ Variable neighbors of factor  $a$  denoted  $N(a)$
- ▶ Factor neighbors of variable  $u$  denoted  $N(u)$
- ▶  $\vec{y}$  is an assignment of values to all variables
- ▶  $u$  takes values from  $\mathcal{Y}_u$
- ▶ If  $S \subseteq V$  is a subset of variables,  $\vec{y}_S$  is a vector of values for variables in  $S$
- ▶  $S$  except variable  $v$  is denoted  $S \setminus v$

## Factor graph [9] (2)

- ▶ E.g.,  $\vec{y}_{N(a)} \in \mathcal{Y}_{N(a)} = \times_{u \in N(a)} \mathcal{Y}_u$  is a cartesian product of domains
- ▶ Associated with each factor  $a$  is a function  $\Psi_a : \mathcal{Y}_{N(a)} \rightarrow \mathbb{R}_+$
- ▶ Joint distribution over  $\vec{y}$  is written in a factored form as  
$$\Pr(\vec{y}) \propto \prod_a \Psi_a(\vec{y}_{N(a)})$$
- ▶ (Will see later how  $\Psi$  can be parameterized)

## Marginals

- ▶ Given a specific value  $\gamma \in \mathcal{Y}_u$  for variable  $u$ , what is the marginal probability  $\Pr(Y_u = \gamma)$ ?
- ▶ If the number of variables is small, this can be enumerated  
$$\sum_{\vec{y} \in \mathcal{Y}_{V \setminus u}} \Pr(Y_{V \setminus u} = \vec{y}, Y_u = \gamma)$$
- ▶ Prohibitive for large  $V$ , factor graph helps us evaluate
- ▶ Converges to exact solution if at most one cycle, effective heuristic otherwise

## Messages

- ▶  $m_{u \rightarrow a}$  is a message from variable  $u$  to factor  $a$
- ▶  $n_{a \rightarrow u}$  is a message from factor  $a$  to variable  $u$
- ▶ Both messages are functions mapping  $\mathcal{Y}_u$  to  $\mathbb{R}_+$
- ▶ I.e., a message is a table
- ▶ For each  $\gamma \in \mathcal{Y}_u$ , nodes  $a$  and  $u$  will send a positive scalar value to each other
- ▶ Usually written as  $m_{u \rightarrow a}(y_u)$  and  $n_{a \rightarrow u}(y_u)$
- ▶ Where  $y_u$  is a specific value like  $\gamma$  that  $Y_u$ , the random variable, can take
- ▶ In step  $k$ ,

$$m_{u \rightarrow a}^{(k)}(y_u) = \prod_{b \in N(u) \setminus a} n_{b \rightarrow u}^{(k-1)}(y_u)$$

$$n_{a \rightarrow u}^{(k)}(y_u) = \sum_{\vec{y} \in \mathcal{Y}_{N(a) \setminus u}} \Psi(\vec{y}, y_u) \prod_{v \in N(a) \setminus u} m_{v \rightarrow a}^{(k)}(y_v)$$

## Messages (2)

- ▶ Note  $y_v$  is bound by the choice of  $\vec{y} \in \mathcal{Y}_{N(a) \setminus u}$
- ▶  $m$  expressed in terms of  $n$  and vice versa
- ▶ Two-pass in specific node order enough for convergence in trees
- ▶ Also converges if the graph has at most one cycle
- ▶ Otherwise no general guarantees; can update factors and variables in any order
- ▶ Called the “sum product” algorithm
- ▶ Superficial resemblance to Kleinberg’s hubs and authorities iterations
- ▶ Python code

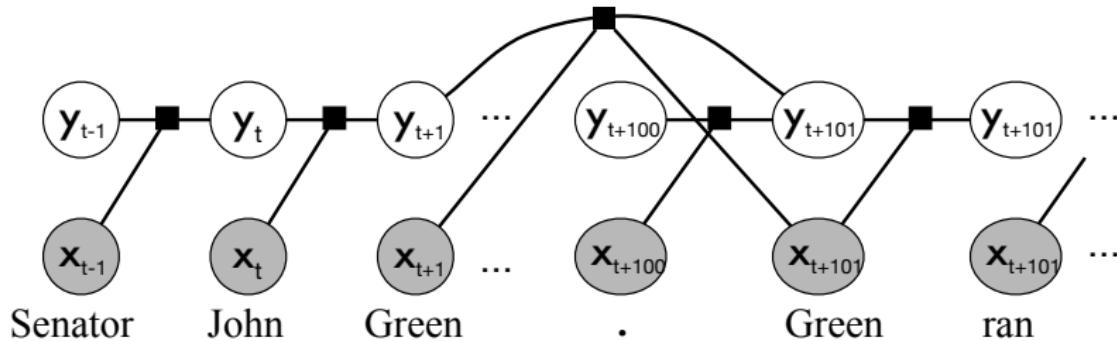
## Max product algorithm

- ▶ Recover single variable marginal as  $\Pr(y_u) \propto \prod_{a \in N(u)} n_{a \rightarrow u}(y_u)$
- ▶ Clearly, different to ask for  $\arg \max_{\vec{y}} \Pr(\vec{y})$  where  $\Pr(\vec{y})$  given as product of factors
- ▶ The latter can be solved by max-product belief propagation:

$$n_{a \rightarrow u}^{(k)}(y_u) = \max_{\vec{y} \in \mathcal{Y}_{N(a) \setminus u}} \Psi(\vec{y}, y_u) \prod_{v \in N(a) \setminus u} m_{v \rightarrow a}^{(k)}(y_v)$$

- ▶ For proofs see books by Koller and Friedman, or Bishop

## Skip-chain CRFs



- ▶ The same word within a document is likely to have the same label
- ▶ Not universally true, can only encourage but not enforce
- ▶ “Green party nominees like John Green did not agree.”
- ▶ Express potential dependencies between distant occurrences  $x_t, y_t$  and  $x_{t'}, y_{t'}$  using a **factor node** ■
- ▶ Can also use factor nodes to couple  $y_t, x_t, y_{t+1}$
- ▶ Factor nodes used to represent near- and far-range coupling

## Parameterizing $\Psi$ in factors

- ▶ Commonly used model

$$\Psi_a(\vec{y}_{N(a)}) = \exp(w_a \cdot \phi(\vec{y}_{N(a)}))$$

where  $\phi$  is a feature function

- ▶ Usually this will result in way too many parameters to train
- ▶ Therefore we **tie** the models at different factors by sharing parameters
- ▶ Represented easily by one level of indirection
- ▶ Let  $\tau(a)$  map from factor  $a$  to a **template**
- ▶ Index  $w$  not with  $a$  but with  $\tau(a)$

$$\Psi_a(\vec{y}_{N(a)}) = \exp(w_{\tau(a)} \cdot \phi(\vec{y}_{N(a)}))$$

- ▶ E.g., in the skip-chain graph,
  - ▶ all factors connecting  $y_{t-1}, x_{t-1}, y_t$  share one set of parameters

## Parameterizing $\Psi$ in factors (2)

- ▶ all factors at the centers of stars like “Green” share a second set of parameters
- ▶ A more common way to write this is using **clique templates**
- ▶ Partition the factor nodes into clusters (also called **cliques**)  $\{C\}$ , and write

$$\Pr(\vec{y}) = \frac{1}{Z} \prod_C \prod_{a \in C} \Psi_a(\vec{y}_{N(a)}) = \frac{1}{Z} \prod_C \prod_{a \in C} e^{w_C \cdot \phi(\vec{y}_{N(a)})}$$

- ▶ Therefore,

$$\log \Pr(\vec{y}) = \sum_C \sum_{a \in C} w_C \cdot \phi(\vec{y}_{N(a)}) - \log Z$$

- ▶ Similar to logistic regression and chain CRF, except that evaluating  $Z$  and expected feature vectors will be via belief propagation

## Training

- ▶ Note that there is no need to represent labeled examples  $\vec{y}^\ell$  separately, because they can be disconnected components in a single graph
- ▶ The training problem is to find  $\{w_C\}$  for all  $C$  so as to maximize

$$\sum_C \sum_{a \in C} w_C \cdot \phi(\vec{y}_{N(a)}) - \log Z$$

- ▶ HW Given a belief propagation subroutine, write down precise pseudocode for computing the objective and gradient for every iteration of a Newton method

## Partially observed data

- ▶ Thus far we have represented all random variables as  $Y_u$  that are unobserved, for simplicity
- ▶ Recall we started with both  $\vec{x}$  and  $\vec{y}$
- ▶ In our information extraction applications some variables are observed as  $x_t$ s
- ▶  $\Pr(\vec{y})$  changes to  $\Pr(\vec{y}|\vec{x})$
- ▶  $Z$  changes to  $Z(\vec{x})$
- ▶  $\phi(\vec{y}_{N(a)})$  changes to  $\phi(\vec{x}, \vec{y}_{N(a)})$
- ▶ Note we can still use any part of  $\vec{x}$

## Dual decomposition

- ▶ Indicator variables  $y(t, m', m) = 1$  if positions  $t - 1, t$  are assigned labels  $m', m$ , 0 otherwise
- ▶ Sequence label space  $\vec{y} \in \mathcal{Y} = \{0, 1\}^{TM^2}$ , size is typically exponential in input size
- ▶ (Not all combinations may be used)
- ▶ Inference amounts to finding  $\arg \max_{\vec{y} \in \mathcal{Y}} h(\vec{y})$
- ▶ For the linear HMM or CRF objective,  $h(\vec{y})$  has the form  $\theta \cdot \vec{y}$ , where  $\theta \in \mathbb{R}^{TM^2}$  depends on observations  $\vec{x}$
- ▶  $\theta$  is set as follows:

$$\theta[t, m', m] = \begin{cases} w \cdot \varphi(x_t, m', m), & \text{for all feasible transitions } m' \rightarrow m \\ -\infty, & \text{for all infeasible transitions} \end{cases}$$

- ▶ For a given  $t$ , only one element in  $y[t, \bullet, \bullet] = 1$ , rest are 0
- ▶ So  $\sum_{m', m} \theta[t, m', m] y[t, m', m] = w \cdot \varphi(x_t, y_{t-1}, y_t)$
- ▶ And  $\theta \cdot \vec{y} = w \cdot \phi(\vec{x}, \vec{y})$

## Dual decomposition (2)

- ▶ HW Why didn't we simply design  $y(t, m) = 1$  if position  $t$  is labeled  $m$  and 0 otherwise? Why pull in  $m'$ ?
- ▶ In what follows, we will use  $x, y$  in place of  $\vec{x}, \vec{y}$
- ▶ Suppose there exists  $\mathcal{Y}' \supset \mathcal{Y}$  such that finding  $\arg \max_{y \in \mathcal{Y}'} \theta \cdot y$  is 'easy'
- ▶ However, the application requires restricting  $\mathcal{Y}'$  in other ways, e.g.,  $\mathcal{Y} = \{y : y \in \mathcal{Y}' \text{ and } Ay = b\}$
- ▶ E.g., restricting labels of some distant tokens to be equal or related can be handled with such constraints
- ▶ Say there are  $c$  constraints
- ▶ Introduce Lagrangian multipliers  $u \in \mathbb{R}^c$  and define

$$L(u, y) = \theta \cdot y + u \cdot (Ay - b)$$

## Dual decomposition (3)

- The dual problem is

$$\min_{u \in \mathbb{R}^c} L(u) = \min_{u \in \mathbb{R}^c} \left\{ \max_{y \in \mathcal{Y}'} L(u, y) \right\}$$

- A 'game' between  $u$  and  $y$ :  $y$  tries to satisfy  $Ay = b$ , if not,  $u$  tries to drag down  $L(u, y)$
- Common approach: set  $u^{(0)} = \vec{0}$ , then:

set 
$$y^{(k)} = \operatorname{argmax}_{y \in \mathcal{Y}'} L(u^{(k-1)}, y)$$

followed by 
$$u^{(k)} = u^{(k-1)} - \delta_k(Ay^{(k-1)} - b)$$

where  $\delta_k$  is a stepsize

## Dual decomposition (4)

- ▶ Note that

$$\begin{aligned}\operatorname{argmax}_{y \in \mathcal{Y}'} L(u^{(k-1)}, y) &= \operatorname{argmax}_{y \in \mathcal{Y}'} \left( \theta \cdot y + u^{(k-1)}(Ay - b) \right) \\ &= \operatorname{argmax}_{y \in \mathcal{Y}'} \theta' \cdot y\end{aligned}$$

which is also ‘easy’

- ▶ Theoretical guarantees in [the paper](#)
- ▶ An extension of the above is where there are **two** overlapping label spaces  $\mathcal{Y}, \mathcal{Z}$ , with

$$f(y) = y \cdot \theta^{(1)} \quad \text{and} \quad g(z) = z \cdot \theta^{(2)}$$

- ▶ E.g., joint sequence labeling consistent with parsing
- ▶ Or joint coref resolution with entity disambiguation

## Dual decomposition (5)

- The decoding objective is

$$\underset{y \in \mathcal{Y}, z \in \mathcal{Z}}{\operatorname{argmax}} y \cdot \theta^{(1)} + z \cdot \theta^{(2)} \quad \text{s.t.} \quad Ay + Cz = b$$

- Same approach as before: relax  $\mathcal{Y}, \mathcal{Z}$  to  $\mathcal{Y}', \mathcal{Z}'$  and define

$$\min_u \max_{y,z} L(u, y, z) = \min_u \left\{ \max_{y,z} y \cdot \theta^{(1)} + z \cdot \theta^{(2)} + u(Ay + Cz - b) \right\}$$

- Set  $u^{(0)} = \vec{0}$ , and alternate

$$y^{(k)}, z^{(k)} = \underset{y,z}{\operatorname{argmax}} L(u^{(k-1)}, y, z)$$

$$\text{with} \qquad u^{(k)} = u^{(k-1)} - \delta_k(Ay^{(k)} + Cz^{(k)} - b)$$

## Dual decomposition (6)

- ▶ Again, note that updating  $y^{(k)}, z^{(k)}$  amounts to separately optimizing

$$\operatorname{argmax}_{y \in \mathcal{Y}'} y \cdot \tilde{\theta}^{(1)} \quad \text{and} \quad \operatorname{argmax}_{z \in \mathcal{Z}'} z \cdot \tilde{\theta}^{(2)}$$

- ▶ Unlike in max-product message passing, at the end of above optimization we still need to **round** the fractional solutions to feasible 0/1 solutions

## Distributional vectors and word clusters

- ▶ Finite labeled data, feature sparsity, and out-of-vocabulary (OOV) words/features have always troubled POS and NER tagging and estimating n-gram statistics
- ▶ E.g. we saw car in the training sequences but never sedan, or Shanghai in test data but only other cities in training data
- ▶ But an unlabeled corpus has enough clues that these are related words
- ▶ A well-established partial fix is **word cluster features**
- ▶ First proposed by IBM researchers in 1992  
<http://aclweb.org/anthology/J/J92/J92-4003.pdf>
- ▶ Given a word like sedan, collect all context windows (say) at most 11 words wide centered on it
- ▶ From these contexts collect a bag of other words, count them
- ▶ Possibly transform from raw counts to TFIDF

## Distributional vectors and word clusters (2)

- ▶ Represent as a sparse vector in a space as large as the corpus vocabulary; perhaps scale to unit length
- ▶ This is the **distributional vector** for sedan
- ▶ Turns out the d.v.'s of similar/related words are similar
- ▶ Can cluster these d.v.'s using standard clustering tools; see [https://en.wikipedia.org/wiki/Brown\\_clustering](https://en.wikipedia.org/wiki/Brown_clustering)
- ▶ In standard CRF implementations, one of the features for each token is its **cluster ID**
- ▶ The best number of clusters may be application-dependent

## Word embeddings [10, 11]

- ▶ In a token window, the **focus** token  $f$  is at the center and others are **context** tokens  $c$
- ▶ Each word in the vocabulary is associated with two embeddings,  $u_w \in \mathbb{R}^D$  as focus and  $v_w \in \mathbb{R}^D$  as context
- ▶ Typically  $D$  ranges from 100 to 1000
- ▶ Two dominant paradigms to train  $\mathbf{U}, \mathbf{V}$

GloVe: 
$$\log X_{fc} \approx u_f \cdot v_c + b_f + b_c,$$

where  $b_w \in \mathbb{R}$  is a per-word offset and  $X_{fc}$  is the cooccurrence count of words  $f$  and  $c$ , and

Word2vec: 
$$\Pr(f, c \text{ cooccur}) = \sigma(u_f \cdot v_c),$$

where  $\sigma(\bullet) = 1/(1 + e^{-\bullet})$  is the sigmoid function

## Word embeddings [10, 11] (2)

- ▶ Variations of low-rank factorization of a transformed cooccurrence matrix
- ▶ Usually only  $\mathbf{U}$  used for downstream tasks, one vector per word, usually scaled to unit L2 norm
- ▶ Although not explicitly trained to those ends, the focus embeddings are useful for many tasks
  - ▶  $u_{\text{auto}} \approx u_{\text{sedan}}$
  - ▶  $u_{\text{king}} - u_{\text{man}} + u_{\text{woman}} \approx u_{\text{queen}}$ , etc.
- ▶ A common use of word embeddings is to inject them into sequential networks as “ $x_t$ ”

## Using word embeddings in sequence labeling

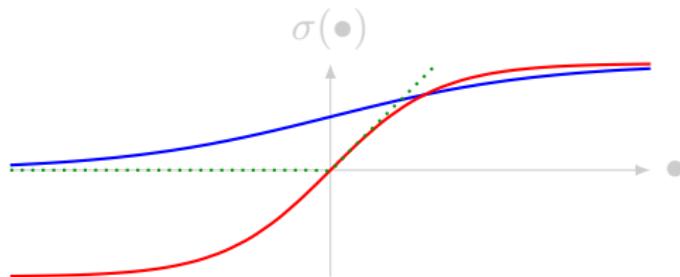
- ▶ Concatenate  $u_{\text{sedan}}$  with all other lexical and derived features of sedan
- ▶ If  $D = 300$  and we had 20000 lexical features and 50 derived features, now we have  $F = 20350$  features
- ▶ (May need some scaling of feature groups so embeddings don't crowd out the high-info derived features)
- ▶ Note, the embeddings themselves do not adapt to the task (POS/NER/etc.) at hand
- ▶ Best current approach:
  - ▶ Directly build neural network to translate from sequence of continuous vectors to sequence of continuous vectors/states
  - ▶ Learn a translation from continuous to discrete states demanded by application
  - ▶ Train through backprop

## Standard neural network terminology

- ▶ A **hidden layer** inputs a vector  $x \in \mathbb{R}^I$ , multiplies by a matrix  $W \in \mathbb{R}^{I \times O}$ , and outputs an output vector  $y = xW \in \mathbb{R}^O$
- ▶ Often a vector  $b \in \mathbb{R}^O$  is added; i.e.,  $y = xW + b$
- ▶ Usually a hidden layer then applies a *nonlinearity*
- ▶ **Nonlinearity**, generically denoted  $\sigma(\bullet)$ , applied elementwise to input scalar, vector, matrix or tensor
  - ▶ **Sigmoid:**  $\sigma(\bullet) = 1/(1 + e^{-\bullet})$
  - ▶ **Rectified linear unit (ReLU):**  $\sigma(\bullet) = \max\{0, \bullet\}$
  - ▶ **Tanh:**  $\sigma(\bullet) = \tanh(\bullet) = \frac{e^\bullet - e^{-\bullet}}{e^\bullet + e^{-\bullet}}$

## Standard neural network terminology (2)

When to use each?



- Sometimes a nonlinearity has its own associated model weights to be learnt, e.g.,

$$\sigma_{a,b}(\bullet) = \frac{1}{1 + \exp(-b(\bullet - a))}$$

which changes the offset ( $a$ ) and slope ( $b$ ) of saturation

- Or it may be folded into the preceding hidden layer by appending a constant dimension to the input vector
- I.e., the typical hidden layer amounts to  $y = \sigma(xW + b)$

## Standard neural network terminology (3)

- ▶ A **softmax** layer turns  $K$  real (positive or negative) numbers  $a_1, \dots, a_K$  into a  $K$ -way multinomial distribution as  
$$b_k = \exp(a_k) / \sum_{k'} \exp(a_{k'})$$
- ▶ (An entropy reduction stage which enhances score skew, and turn any set of numbers into a multinomial distribution)
- ▶ Typically used for transforming the output of a hidden layer into a multinomial distribution over labels
- ▶ Tensorflow hello world: `regression.py`, `svm.py`
- ▶ Variables, placeholders, loss expression, optimizer, session, tensorboard
- ▶ Keras: higher level constructs like fully connected layer, convolutional network, sequence model, ...

## Sentence and passage representation

- ▶ From individual word embeddings to representation of sentences and passages
- ▶ Sample application: matching query with candidate passage
  - ▶ Cart before horse  $\neq$  horse before cart
- ▶ Common neural network styles

**Baseline 0:** Simple symmetric aggregate (sum, average) of word vectors

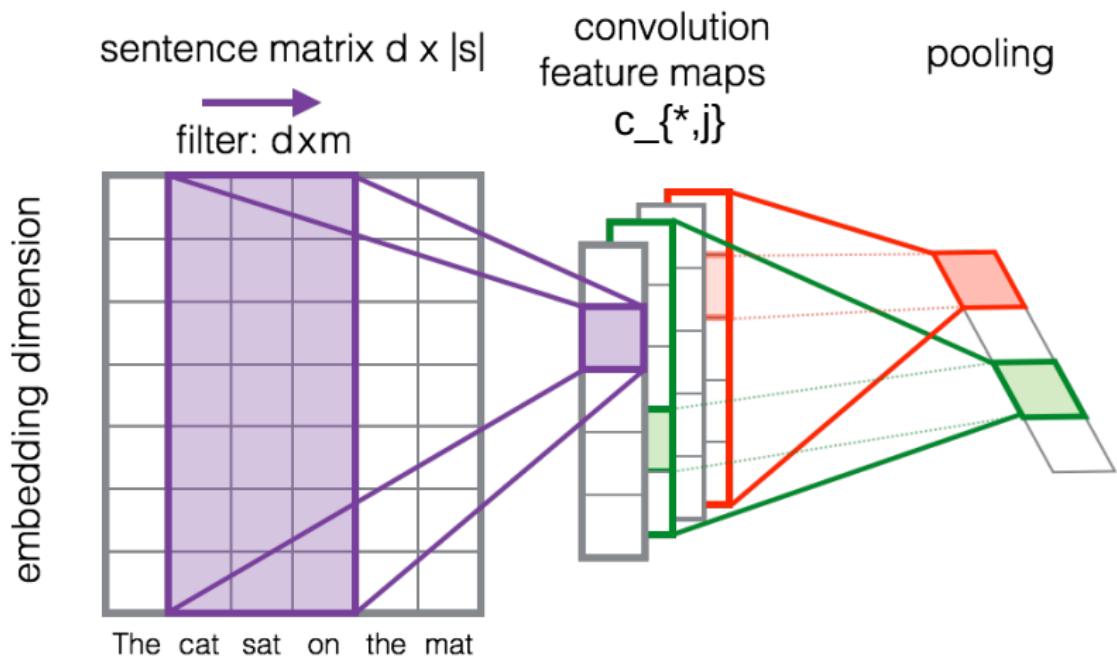
**ConvNets:** Hierarchical position-aware aggregation without parsing

**Recursive networks:** Ditto, along parse structure provided by a separate parser (say, PCFG-based)

**Recurrent networks:** Deep counterpart of chain CRFs: ordered updates to latent continuous state vector

# Convolutional networks over text

- ▶ Composing continuous representation without parses
- ▶ Understanding convnets for NLP



## Convolutional networks over text (2)

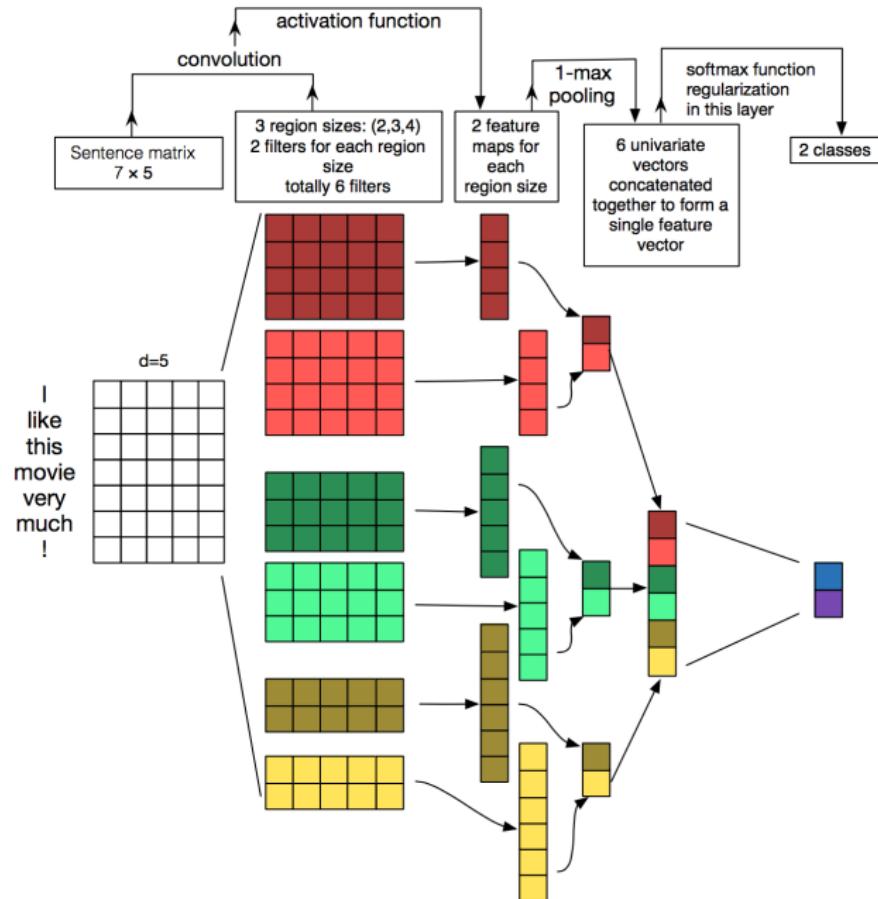
- ▶  $d \times |s|$  word embeddings matrix  $\mathbf{S}$  for sentence with  $|s|$  words, here  $7 \times 6$
- ▶ In the figure,  $7 \times 3$  stencil can slide horizontally through four positions
- ▶ Weight matrix is  $\mathbf{W} \in \mathbb{R}^{7 \times 3}$
- ▶ When left edge of  $\mathbf{W}$  is offset to column  $k$  of  $\mathbf{S}$ , we compute

$$c_k = \sum_{i=0}^6 \sum_{j=0}^2 W[i, j] S[i, j + k]$$

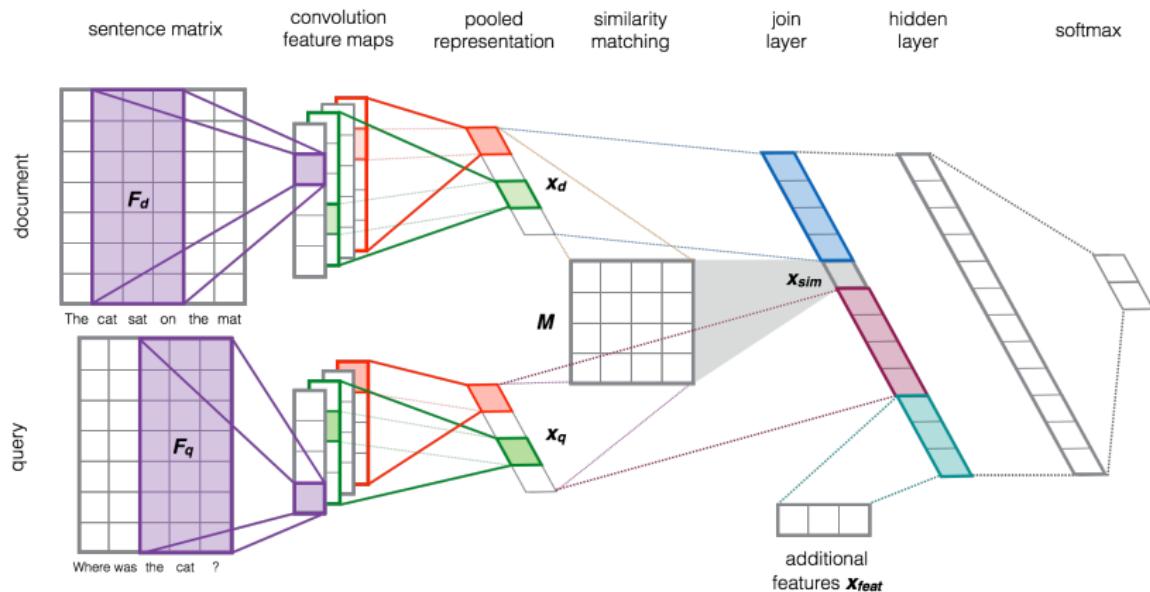
- ▶ This gives  $\mathbf{c}$ , a vector with 4 numbers corresponding to  $k = 0, \dots, 3$
- ▶ Now in place of one  $\mathbf{W}$ , use  $\mathbf{W}, \mathbf{W}, \mathbf{W}, \mathbf{W}, \mathbf{W}$
- ▶ Leading to  $\mathbf{c}, \mathbf{c}, \mathbf{c}, \mathbf{c}, \mathbf{c}$ ; i.e., a matrix  $\mathbf{C} \in \mathbb{R}^{4 \times 5}$

## Convolutional networks over text (3)

- ▶ More weights: a vector  $\mathbf{u}$ , leading to vector  $\mathbf{f} = \mathbf{C}\mathbf{u} \in \mathbb{R}^4$  as shown
- ▶ This gives a 4-dimensional vector representation of the passage
- ▶ Can do (at least) two kinds of things with  $\mathbf{f}$ :
  - ▶ Learn a classifier whose input is  $\mathbf{f}$ , to predict labels like topic, sentiment, product rating: if there are  $M$  labels, use matrix  $\mathbf{V} \in \mathbb{R}^{4 \times M}$  and output label scores  $\text{softmax}(\mathbf{f}\mathbf{V})$
  - ▶ Compare two passages for 'semantic' similarity or entailment ( $s_1 \implies s_2$ ): details later
- ▶ If unsure about best shape of sliding stencil, use many of various sizes



# Comparing two passages



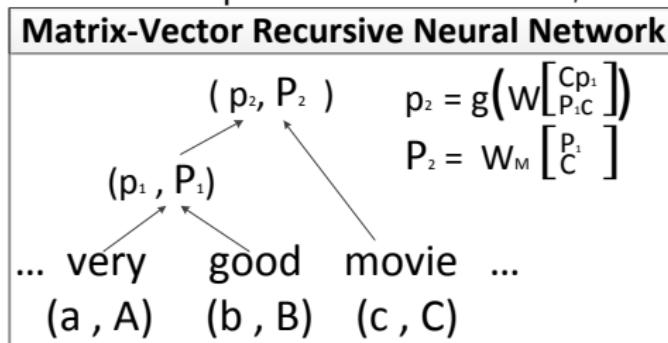
- ▶ E.g., to score passages in response to queries
- ▶ Process query and passage through “Siamese” (twin) networks with tied model weights
- ▶ Results in representations  $x_{\text{doc}}$  and  $x_{\text{query}}$
- ▶ Compare these using another network

## Composition along parse trees [13]

- ▶ If we have a reliable dependency or constituency parse, makes sense to use it for composing word vectors
- ▶ Words  $a, b$ , with vectors  $w_a, w_b \in \mathbb{R}^D$ , are children of internal node  $c$
- ▶ First-cut: Fit  $w_c = \sigma \left( M \begin{bmatrix} w_a \\ w_b \end{bmatrix} \right)$  where  $M \in \mathbb{R}^{D \times 2D}$
- ▶ Single  $M$  unlikely to be able to capture all natural language 'operators'
- ▶ Associate each word  $a$  with vector  $w_a \in \mathbb{R}^D$  and matrix  $M_a \in \mathbb{R}^{D \times 2D}$
- ▶ Let  $w_c = \sigma \left( \textcolor{red}{M_0} \begin{bmatrix} M_b w_a \\ M_a w_b \end{bmatrix} \right)$  where  $M_0 \in \mathbb{R}^{D \times 2D}$  is a global matrix

## Composition along parse trees [13] (2)

- ▶ Each word operates on the other, results combined



- ▶ For compositionality, must define  $M_c$  as well:  $M_c = M_1 \begin{bmatrix} M_a \\ M_b \end{bmatrix}$  where  $M_1 \in \mathbb{R}^{D \times 2D}$  is another global matrix
- ▶ Overall loss depends on task at hand
  - ▶  $w_{\text{root}}$  can be mapped to softmax( $M_{\text{root}} w_{\text{root}}$ ) for classification tasks
  - ▶ IMDB movie reviews: unbelievably sad, really awesome, pretty bad, not terrible

## Composition along parse trees [13] (3)

Method	Avg KL
$\frac{1}{2}(w_a + w_b)$	0.103
$w_a \circ w_b$	0.103
$M_a w_b$	0.103
MV-RNN	<b>0.091</b>

- ▶ Classifying semantic relationships
- ▶ “My [apartment]<sub>e1</sub> has a pretty large [kitchen]<sub>e2</sub>” → component-whole

Relationship	Sentence with labeled nouns for which to predict relationships
Cause-Effect(e2,e1)	Avian [influenza] <sub>e1</sub> is an infectious disease caused by type a strains of the influenza [virus] <sub>e2</sub> .
Entity-Origin(e1,e2)	The [mother] <sub>e1</sub> left her native [land] <sub>e2</sub> about the same time and they were married in that city.
Message-Topic(e2,e1)	Roadside [attractions] <sub>e1</sub> are frequently advertised with [billboards] <sub>e2</sub> to attract tourists.
Product-Producer(e1,e2)	A child is told a [lie] <sub>e1</sub> for several years by their [parents] <sub>e2</sub> before he/she realizes that ...
Entity-Destination(e1,e2)	The accident has spread [oil] <sub>e1</sub> into the [ocean] <sub>e2</sub> .
Member-Collection(e2,e1)	The siege started, with a [regiment] <sub>e1</sub> of lightly armored [swordsmen] <sub>e2</sub> ramming down the gate.
Instrument-Agency(e2,e1)	The core of the [analyzer] <sub>e1</sub> identifies the paths using the constraint propagation [method] <sub>e2</sub> .
Component-Whole(e2,e1)	The size of a [tree] <sub>e1</sub> [crown] <sub>e2</sub> is strongly correlated with the growth of the tree.
Content-Container(e1,e2)	The hidden [camera] <sub>e1</sub> , found by a security guard, was hidden in a business card-sized [leaflet box] <sub>e2</sub> placed at an unmanned ATM in Tokyo's Minato ward in early September.

## Composition along parse trees [13] (4)

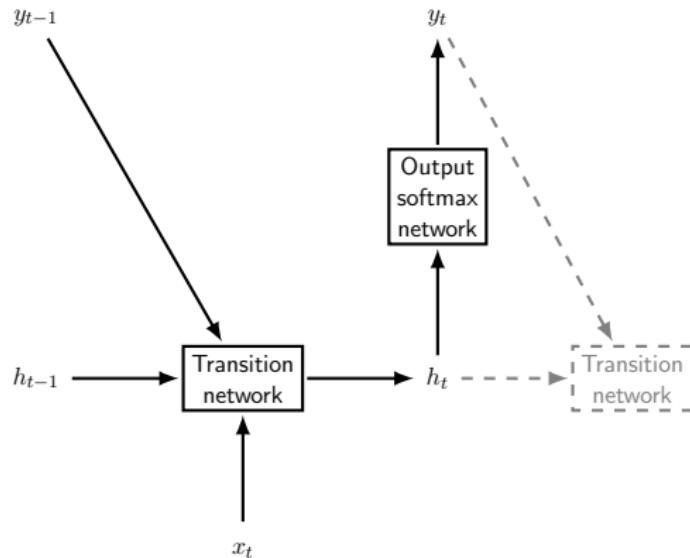
### ► Performance on SemEval-2010, task 8

Classifier	Feature Sets	F1
SVM	POS, stemming, syntactic patterns	60.1
SVM	word pair, words in between	72.5
SVM	POS, WordNet, stemming, syntactic patterns	74.8
SVM	POS, WordNet, morphological features, thesauri, Google $n$ -grams	77.6
MaxEnt	POS, WordNet, morphological features, noun compound system, thesauri, Google $n$ -grams	77.6
SVM	POS, WordNet, prefixes and other morphological features, POS, dependency parse features, Levin classes, PropBank, FrameNet, NomLex-Plus, Google $n$ -grams, paraphrases, Textrunner	82.2
RNN	-	74.8
Lin.MVR	-	73.0
MV-RNN	-	79.1
RNN	POS,WordNet,NER	77.6
Lin.MVR	POS,WordNet,NER	78.7
MV-RNN	POS,WordNet,NER	<b>82.4</b>

Table 4: Learning methods, their feature sets and F1 results for predicting semantic relations between nouns. The MV-RNN outperforms all but one method without any additional feature sets. By adding three such features, it obtains state of the art performance.

## Basic sequence network template

- ▶ Discrete state  $y_t$  replaced with continuous vector states  $h_t$ ; usually  $h_0 = \vec{0}$
- ▶ The transition network inputs previous state  $h_{t-1}$  and current input  $x_t$  (also continuous form) and outputs current state  $h_t$
- ▶  $y_t$  for end application derived from  $h_t$  using an output network ending in a softmax



## Basic recurrent network (RNN)

- ▶ The transition network and the output network are standard multi-layer neural networks with linear combinations and nonlinearities at every hidden layer

Elman network: 
$$h_t = \sigma(x_t W_h + h_{t-1} U_h + b_h)$$

$$y_t = \sigma(h_t W_y + b_y)$$

Jordan network: 
$$h_t = \sigma(x_t W_h + y_{t-1} U_h + b_h)$$

$$y_t = \sigma(h_t W_y + b_y)$$

- ▶ Weights  $W_h, U_h, b_h$  of transition networks at all positions are tied; same for weights  $W_y, b_y$  in the output network
- ▶ Stream of consciousness
- ▶ Potential problems: vanishing or exploding gradient
  - ▶ I grew up in France ... I speak fluent French

## Gated recurrent unit (GRU) [14]

Reset	$r_t = \sigma(x_t W_r + h_{t-1} U_r + b_r)$
Combine	$c_t = \tanh(x_t W_h + (r_t \circ h_{t-1}) U_h + b_h)$
Update	$z_t = \sigma(x_t W_z + h_{t-1} U_z + b_z)$
Output	$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ c_t$

- ▶  $\sigma$  is sigmoid; other nonlinearities specified explicitly
- ▶  $r_t$  decides if  $c_t$  is computed as in a basic RNN, or  $h_{t-1}$  is discarded and only  $x_t$  is used
- ▶  $z_t$  decides how to blend  $c_t$  and  $h_{t-1}$  as-is
- ▶ If we set by force  $r_t \approx 1$  and  $z_t \approx 1$ , equivalent to basic RNN
- ▶ <https://arxiv.org/abs/1412.3555>

# Long short-term memory (LSTM) [15]

In	$i_t = \sigma(x_t U_i + h_{t-1} W_i + b_i)$
Forget	$f_t = \sigma(x_t U_f + h_{t-1} W_f + b_f)$
Out	$o_t = \sigma(x_t U_o + h_{t-1} W_o + b_o)$
RNN	$g_t = \tanh(x_t U_g + h_{t-1} W_g + b_g)$
Memory	$c_t = c_{t-1} \circ f_t + g_t \circ i_t$
State	$h_t = \tanh(c_t) \circ o_t$

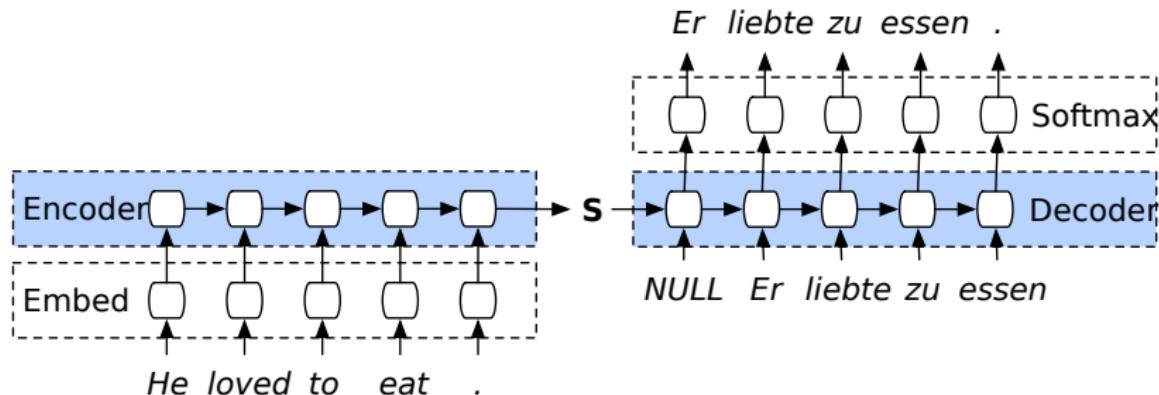
- ▶ Understanding LSTMs
- ▶ Accuracy comparable to GRUs
- ▶ Widely used for sequence labeling, sequence-to-sequence comparison, sequence-to-sequence translation, etc.
- ▶ Usually left-to-right and right-to-left LSTMs in tandem as a **bi-LSTM**; captures context from both sides
- ▶ Unreasonable effectiveness

## RNNs as language models and translators

- ▶ Suppose input is word sequence ( $x_t$ )
- ▶ Whose embeddings are inputs to an RNN
- ▶ The label sequence ( $y_t$ ) can be defined as looking ahead one step in the same word sequence, i.e., ( $x_{t+1}$ )
- ▶ E.g., having read “I speak fluent       ”, what next word do we expect?
- ▶ Thus, an unsupervised corpus can lead to loads of supervised “ $y_t$ ” label data

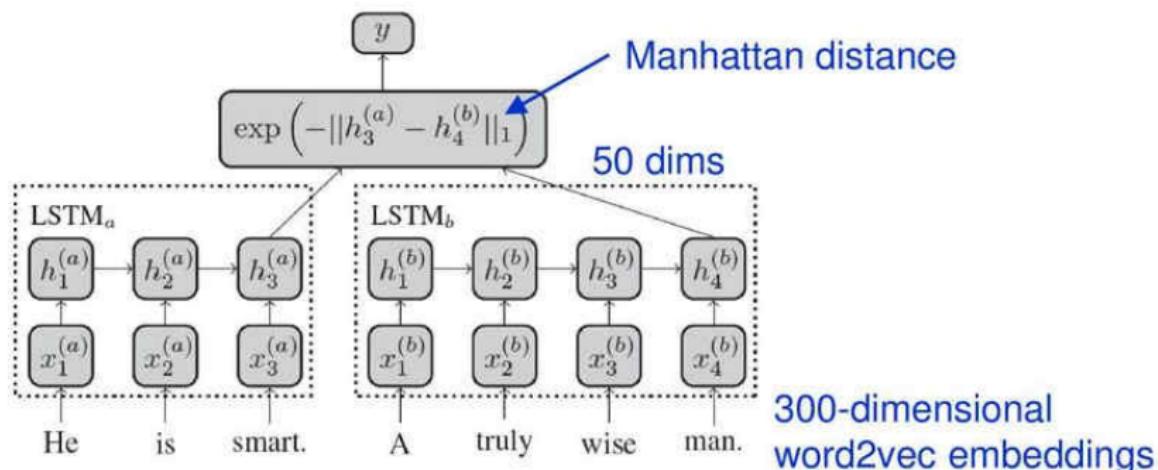
## RNNs as language models and translators (2)

- ▶ Also works for translation



- ▶ First encode the English sentence
- ▶ Ending with a special marker ‘.’
- ▶ “Meaning” of English sentence captured in  $S$
- ▶ Send in  $NULL$ , starting the German decoder

# Does one sentence imply another?



Quotable expletives and discontent:

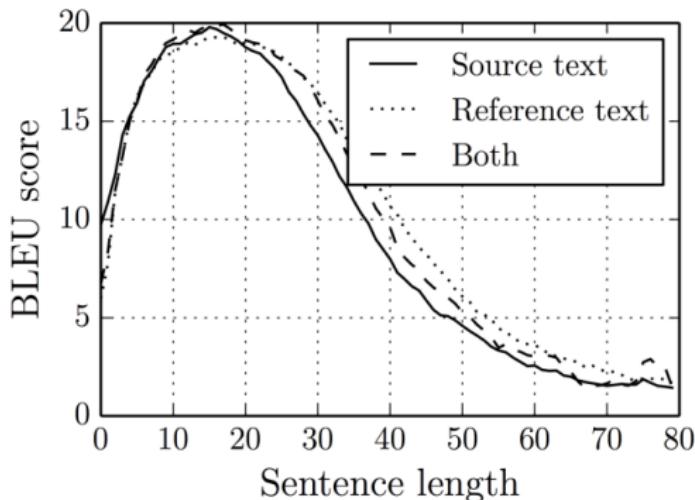
“You can't cram the meaning of a whole %&!\$# sentence into a single \$&!#\* vector!” — Ray Mooney (2014)

“The pooling operation used in convolutional neural networks is a big mistake and the fact that it works so well is a disaster.”

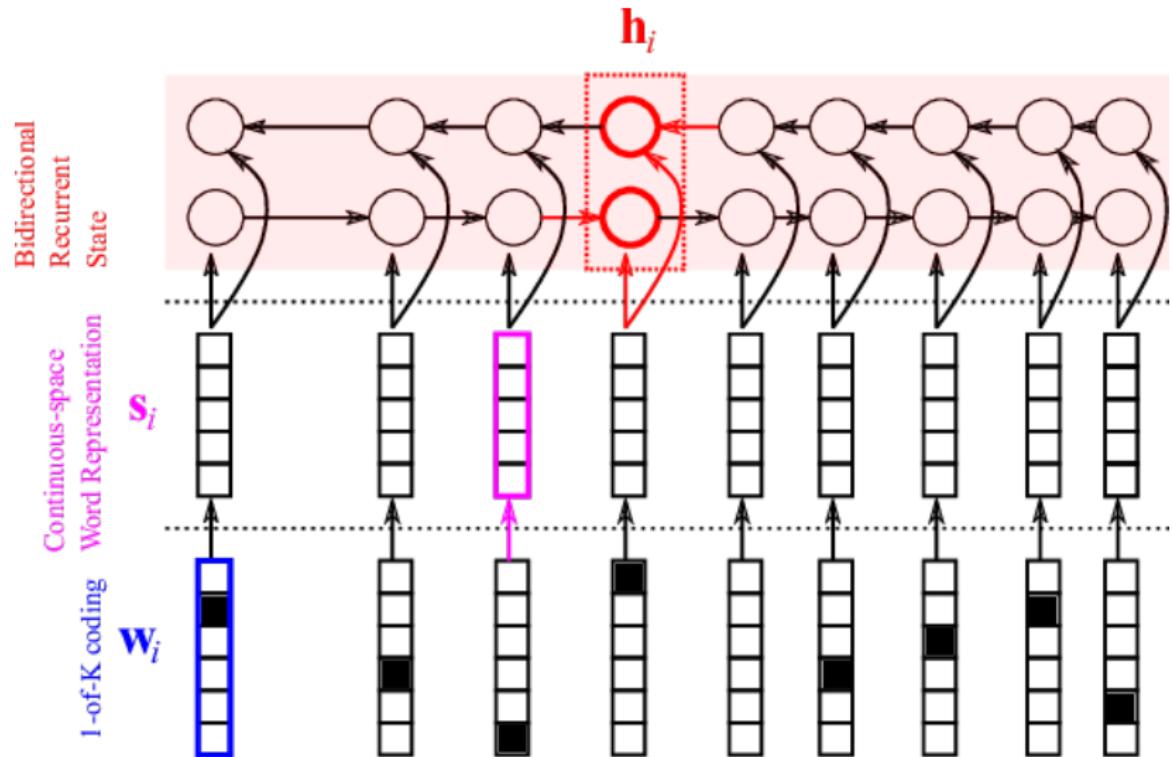
— Geoffrey Hinton (2014)

## Fixed size passage encoding

- ▶ Fixed passage encoding dimension independent of passage length is a problem
- ▶ **Attention** to the rescue

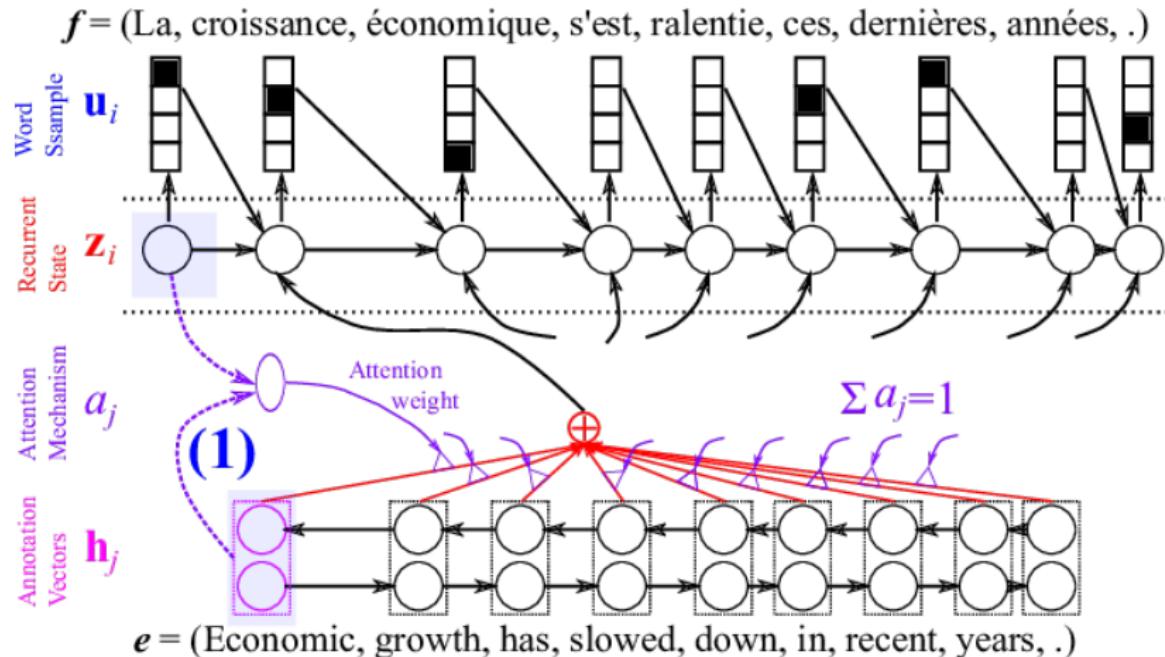


# Encoding the English sentence



$e = (\text{Economic, growth, has, slowed, down, in, recent, years, .})$

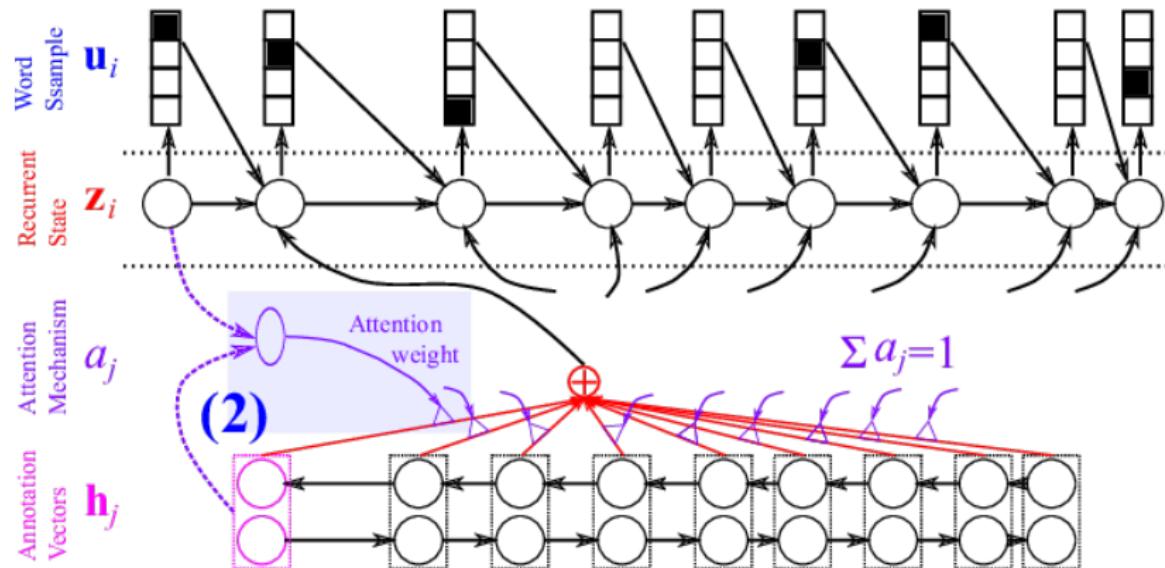
# Evaluating attention from target position [16]



- ▶ For each target position  $i$
- ▶ Choose one source position  $j$
- ▶ Whose state  $h_j$ , along with  $z_{i-1}$  informs decoder

# Computing attention-weighted average English state

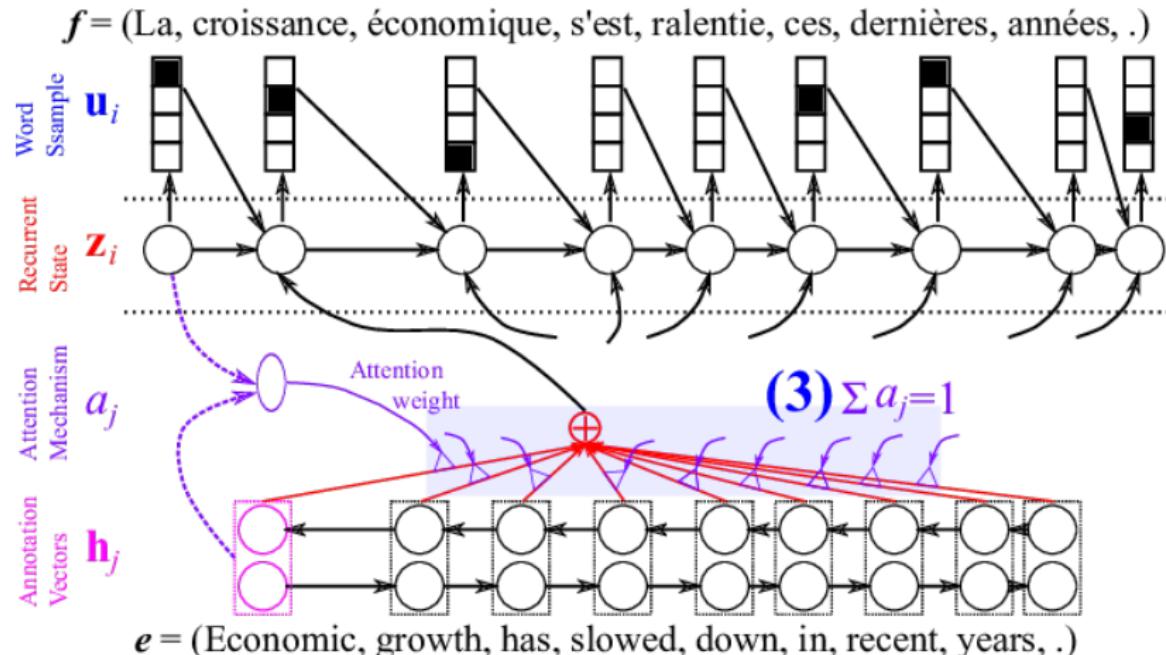
$f = (\text{La, croissance, économique, s'est, ralenti, ces, dernières, années, .})$



$e = (\text{Economic, growth, has, slowed, down, in, recent, years, .})$

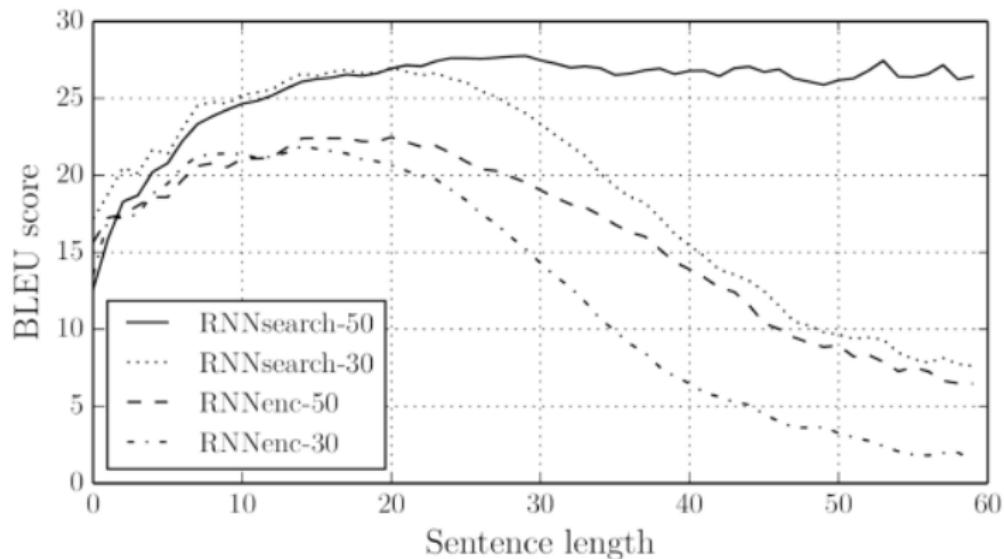
- ▶ Choose  $h_j$  in a soft manner to support loss backprop
- ▶ Compute  $a_j$  as some network function of  $h_j$  and  $z_i$
- ▶ Normalize using softmax to add up to 1 and increase skew
- ▶ Compute  $\sum_j a_j h_j$  and feed into decoder's  $(i + 1)$ th cell

## Emitting next French word



- ▶ Decoder network itself is not bidirectional
- ▶ Both  $i$  and  $i'$  can choose a common  $j^*$
- ▶ But each  $i$  can (soft-) select only one  $j$

## Improved BLEU score



Paper

## Attention strength examples

Economic growth has slowed down in recent years .



Das Wirtschaftswachstum hat sich in den letzten Jahren verlangsamt .

Economic growth has slowed down in recent years .



La croissance économique s' est ralentie ces dernières années .

- ▶ *Wirtschafts* means *economic*
- ▶ *Wirtschaftswachstum* means *economic growth*
- ▶ Note the missing line from *growth*
- ▶ Also, *dernières* means *latest*, faint link to *recent*
- ▶ Should allow many-to-many links in general

## Quick summary of using neural networks for text

- ▶ Start with individual word embeddings
- ▶ ConvNets provide somewhat position-sensitive aggregation of words of a sentence or passage, without demanding hard parses
- ▶ If parses are available, can compose word embeddings along the parse structure
- ▶ (Not all compositions are valid, e.g., “New York”, “rat race”, “red carpet”, “snake oil”)
- ▶ Counterpart to chain CRFs is the RNN family
- ▶ Exploding and vanishing gradients
- ▶ Addressed by GRU and LSTM
- ▶ Applied in parsing, entailment testing, machine translation, question answering, ...
- ▶ Various attention mechanisms boost performance

## References

- [1] D. Freitag and A. McCallum, "Information extraction using HMMs and shrinkage," in *Papers from the AAAI-99 Workshop on Machine Learning for Information Extraction*, 1999, pp. 31–36.
- [2] V. R. Borkar, K. Deshmukh, and S. Sarawagi, "Automatic text segmentation for extracting structured records," in *SIGMOD Conference*, 2001. [Online]. Available:  
<http://www.it.iitb.ac.in/~sunita/papers/sigmod01.pdf>
- [3] S. Sarawagi, "Information extraction," *FnT Databases*, vol. 1, no. 3, 2008. [Online]. Available:  
<http://www.cse.iitb.ac.in/~sunita/papers/ieSurvey.pdf>
- [4] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *ICML*, 2001, pp. 282–289. [Online]. Available: [ftp://ftp.cis.upenn.edu/pub/datamining/public\\_html/ReadingGroup/papers/crf.pdf](ftp://ftp.cis.upenn.edu/pub/datamining/public_html/ReadingGroup/papers/crf.pdf)

## References (2)

- [5] I. Tschantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *JMLR*, vol. 6, no. Sep., pp. 1453–1484, 2005. [Online]. Available: <http://ttic.uchicago.edu/~altun/pubs/TsoJoaHofAlt-JMLR.pdf>
- [6] F. Sha and F. Pereira, "Shallow parsing with conditional random fields," in *HLT-NAACL*, 2003, pp. 134–141. [Online]. Available: <http://acl.ldc.upenn.edu/N/N03/N03-1028.pdf>
- [7] S. Sarawagi and W. W. Cohen, "Semi-markov conditional random fields for information extraction," in *NIPS Conference*, 2005, pp. 1185–1192. [Online]. Available: <http://papers.nips.cc/paper/2648-semi-markov-conditional-random-fields-for-information-extraction.pdf>
- [8] P. Viola and M. Narasimhan, "Learning to extract information from semi-structured text using a discriminative context free grammar," in *SIGIR Conference*, 2005, pp. 330–337. [Online]. Available: <https://dl.acm.org/citation.cfm?id=1076091>

## References (3)

- [9] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. [Online]. Available: <http://mitpress.mit.edu/catalog/author/default.asp?aid=15828>
- [10] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS Conference*, 2013, pp. 3111–3119. [Online]. Available: <https://goo.gl/x3DTzS>
- [11] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation." in *EMNLP Conference*, vol. 14, 2014, pp. 1532–1543. [Online]. Available: <http://www.emnlp2014.org/papers/pdf/EMNLP2014162.pdf>
- [12] A. Severyn and A. Moschitti, "Learning to rank short text pairs with convolutional deep neural networks," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2015, pp. 373–382. [Online]. Available: <http://disi.unitn.it/~severyn/papers/sigir-2015-long.pdf>

## References (4)

- [13] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *EMNLP Conference*, 2012, pp. 1201–1211. [Online]. Available: [http://www.socher.org/uploads/Main/SocherHuvalManningNg\\_EMNLP2012.pdf](http://www.socher.org/uploads/Main/SocherHuvalManningNg_EMNLP2012.pdf)
- [14] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014. [Online]. Available: <https://arxiv.org/pdf/1412.3555.pdf>
- [15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997. [Online]. Available: <https://www.mitpressjournals.org/doi/pdfplus/10.1162/neco.1997.9.8.1735>
- [16] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2014. [Online]. Available: <http://arxiv.org/abs/1409.0473>

From coarse NER to fine types

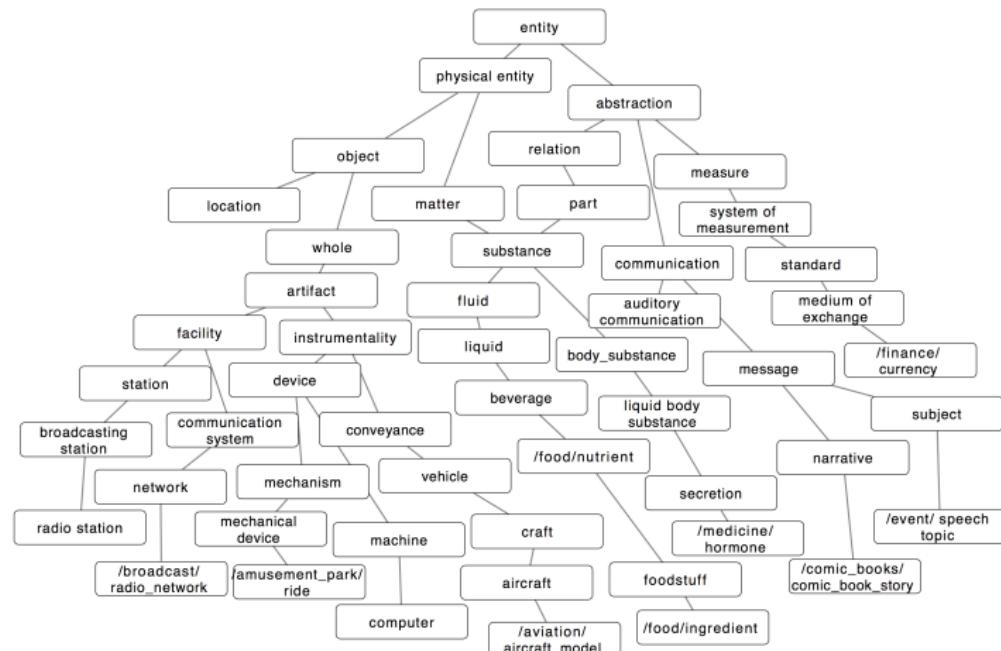
# FIGER type catalog (112 fine types) [1]

<b>person</b>	doctor	<b>organization</b>	terrorist_organization
actor	engineer	airline	government_agency
architect	monarch	company	government
artist	musician	educational_institution	political_party
athlete	politician	fraternity_sorority	educational_department
author	religious_leader	sports_league	military
coach	soldier	sports_team	news_agency
director	terrorist		
<b>location</b>	body_of_water	<b>product</b>	<b>art</b>
city	island	camera	written_work
country	mountain	mobile_phone	film
county	glacier	computer	newspaper
province	astral_body	software	music
railway	cemetery	game	
road	park	instrument	<b>event</b>
bridge		weapon	military_conflict
			attack
			natural_disaster
			election
			sports_event
			protest
			terrorist_attack
<b>building</b>	time	chemical_thing	website
airport	color	biological_thing	broadcast_network
dam	award	medical_treatment	broadcast_program
hospital	educational_degree	disease	tv_channel
hotel	title	symptom	currency
library	law	drug	stock_exchange
power_station	ethnicity	body_part	algorithm
restaurant	language	living_thing	programming_language
sports_facility	religion	animal	transit_system
theater	god	food	transit_line

# Google fine type (GFT) catalog [2]

PERSON	LOCATION	ORGANIZATION	OTHER
<b>artist</b> actor author director music	<b>structure</b> airport government hospital hotel restaurant sports facility theatre	<b>company</b> broadcast news	<b>art</b> broadcast film music stage writing
<b>education</b> student teacher	<b>geography</b> body of water island mountain	<b>education</b> <b>government</b> <b>military</b> <b>music</b> <b>political party</b> <b>sports league</b> <b>sports team</b> <b>stock exchange</b> <b>transit</b>	<b>event</b> accident election holiday natural disaster protest sports event violent conflict
<b>athlete</b> <b>business</b> <b>coach</b> <b>doctor</b> <b>legal</b> <b>military</b> <b>political figure</b> <b>religious leader</b> <b>title</b>	<b>transit</b> bridge railway road	<b>health</b> malady treatment	<b>language</b> programming language
	<b>celestial</b> city country park	<b>award</b> <b>body part</b> <b>currency</b>	<b>living thing</b> animal
			<b>product</b> camera car computer mobile phone software weapon
			<b>food</b> <b>heritage</b> internet <b>legal</b> <b>religion</b> <b>scientific</b> <b>sports &amp; leisure</b> <b>supernatural</b>

# TypeNet catalog [3]



- ▶ 1081 Freebase types
- ▶ Connected to 860 WordNet types
- ▶ Deep hierarchy with average depth of 7.8

## Fine type tagging: Motivation

- ▶ Suppose John Smith is a cricket player not yet in Wikipedia
- ▶ But mentioned in local news about county cricket
- ▶ Query is “**Which cricketer** took four wickets in one over last year against Birmingham?”
- ▶ Potential evidence passage: “**Birmingham** crashed out of the match after losing **four wickets** to **Smith** in a single **over**.”
- ▶ Goal is to collect John Smith as a (strong) candidate, for which we must know that **Smith** refers to a cricketer
- ▶ Experience suggests [4, 5] (thousands of) finer types better for QA than (hundreds of) fine types, but hard to infer from query or context

## Approaches

- ▶ Feature engineering (bag of features) and SVM or logistic regression [1, 2]
- ▶ Embed class labels as well as features — handles similar or overlapping types [6]
- ▶ Sequence models (LSTMs) [7]
- ▶ Attention mechanism to mitigate training noise [8]

## Type tagging: basic idea

- ▶ Suppose we had labeled training data
  - ▶ "Nobody scored as many goals in one match as █ in 2004."
  - ▶ Type label of █ is /person/athlete
- ▶ (How to get labeled data? Will return to this)
- ▶ Collect features from mention context
  - ▶ Context words: scored, goals, match
  - ▶ Word bigrams, shapes ("Aa" for "Ohio")
  - ▶ Dependency paths<sup>3</sup> connecting █ to context words
  - ▶ Discourse topic (sports, politics)
- ▶ Train a standard multi-class multi-label classifier
  - ▶ Could regard context features as bags
  - ▶ Or sequences for more accuracy — usually LSTMs
- ▶ I.e., █ in a given context can be assigned multiple types
  - ▶ Needed if types are not mutually exclusive
  - ▶ E.g., a person can be president and author in the same context

---

<sup>3</sup> "Washington sat on his favorite Barcelona and popped open a Newcastle."

# FIGER system and features

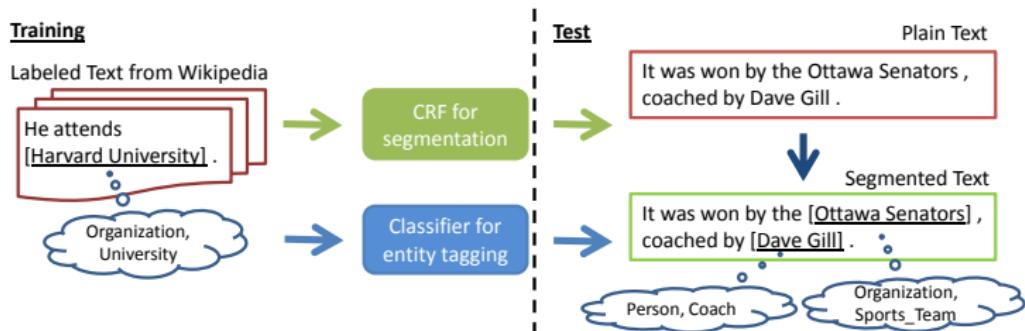


Figure 1: System architecture of FIGER.

Feature	Description	Example
Tokens	The tokens of the segment.	"Eton"
Word Shape	The word shape of the tokens in the segment.	"Aa" for "Eton" and "A0" for "CS446".
Part-of-Speech tags	The part-of-speech tags of the segment.	"NNP"
Length	The length of the segment.	1
Contextual unigrams	The tokens in a contextual window of the segment.	"victory", "for", "
Contextual bigrams	The contextual bigrams including the segment.	"victory for", "for Eton" and "Eton ."
Brown clusters	The cluster id of each token in the segment (using the first 4, 8 and 12-bit prefixes).	"4_1110", "8_11100111", etc.
Head of the segment	The head of the segment following the rules by Collins (1999).	"HEAD_Eton"
Dependency	The Stanford syntactic dependency (De Marneffe, MacCartney, and Manning 2006) involving the head of the segment.	"prep_for:seal:dep"
ReVerb patterns	The frequent lexical patterns as meaningful predicates	"seal.victory.for:dep"

C. J. Ottaway scored his celebrated 108 to seal victory for Eton.

# Google fine types and baseline system [2]

PERSON	LOCATION	ORGANIZATION	OTHER
<b>artist</b> actor author director music	<b>structure</b> airport government hospital hotel restaurant sports facility theatre	<b>company</b> broadcast news	<b>art</b> broadcast film music stage writing
<b>education</b> student teacher	<b>geography</b> body of water island mountain	<b>education</b> <b>government</b> <b>military</b> <b>music</b> <b>political party</b> <b>sports league</b> <b>sports team</b> <b>stock exchange</b> <b>transit</b>	<b>event</b> accident election holiday natural disaster protest sports event violent conflict
<b>athlete</b> <b>business</b> <b>coach</b> <b>doctor</b> <b>legal</b> <b>military</b> <b>political figure</b> <b>religious leader</b> <b>title</b>	<b>transit</b> bridge railway road	<b>health</b> malady treatment	<b>product</b> camera car computer mobile phone software weapon
	<b>celestial</b> city country park	<b>award</b> <b>body part</b> <b>currency</b>	<b>food</b> <b>heritage</b> <b>internet</b> <b>legal</b> <b>religion</b> <b>scientific</b> <b>sports &amp; leisure</b> <b>supernatural</b>

- ▶ Minor tweaks to FIGER types
- ▶ Improvements in collecting labeled data

## Google fine types and baseline system [2] (2)

- ▶ Enhanced classification
- ▶ Training data expected to have extraneous labels
  - ▶ Entity Obama is-a politician, (ex-) POTUS, lawyer, book author, parent, ...
  - ▶ In a given context, one or few types may be ‘active’
  - ▶ But training instance produced with all type labels
- ▶ To mitigate problems from extraneous labels, use weighted approximate rank pairwise (**WARP**) loss
- ▶ Features<sup>4</sup> (e.g. for “... who Barack H. Obama first picked . . . ”)

Feature	Description	Example
Head	The syntactic head of the mention phrase	“Obama”
Non-head	Each non-head word in the mention phrase	“Barack”, “H.”
Cluster	Word cluster id for the head word	“59”
Characters	Each character trigram in the mention head	“:ob”, “oba”, “bam”, “ama”, “ma:”
Shape	The word shape of the words in the mention phrase	“Aa A. Aa”
Role	Dependency label on the mention head	“subj”
Context	Words before and after the mention phrase	“B:who”, “A:first”
Parent	The head’s lexical parent in the dependency tree	“picked”
Topic	The most likely topic label for the document	“politics”

---

<sup>4</sup> “Washington sat on his favorite Barcelona and opened a Newcastle.”

## Embedding type labels with WARP loss [6]

- ▶ Mention contexts represented as  $x$
- ▶ A common situation is  $x \in \mathbb{R}^D$ , for which we choose embedding  $f(x) = \mathbf{A}x \in \mathbb{R}^H$ , where  $\mathbf{A} \in \mathbb{R}^{H \times D}$
- ▶ Want to exploit related types by embedding each type to a vector; similar types expected to embed to similar vectors
- ▶ Let  $\delta_t$  is the 1-hot vector for  $t$
- ▶ Let the  $t$ th column of matrix  $\mathbf{B} \in \mathbb{R}^{H \times T}$  represent the  $H$ -dimensional embedding of type  $t$
- ▶ I.e., we can use notation  $g(t) = \mathbf{B}\delta_t$  as the embedding  $g(t) \in \mathbb{R}^H$
- ▶ The score of a single type label  $t$  for context  $x$  is  $s_t(x) = f(x) \cdot g(\delta_t)$
- ▶ Multiple type labels may be valid in both train and test instances

## Embedding type labels with WARP loss [6] (2)

- ▶  $i$ th labeled instance is  $(x_i, \mathbf{y}_i)$  where  $\mathbf{y}_i$  represents a label set, possibly as a few-hot vector in  $\{0, 1\}^T$
- ▶ Exact inference must explore all  $2^T$  label subsets:  
$$\hat{\mathbf{y}} = \operatorname{argmax}_{\mathbf{y}} f(x) \cdot g(\mathbf{y})$$
- ▶ To avoid high inference cost, cast as label **ranking**
- ▶ Overall score vector  $\mathbf{s}(x) = (\dots, s_t(x), \dots) \in \mathbb{R}^T$
- ▶ Goal is to rank all correct labels before any incorrect one
- ▶ Loss on instance  $x_i, \mathbf{y}_i$  is some function of the rank(s) of the correct label(s) in list of types sorted by decreasing score
- ▶ Let  $\operatorname{rank}(t, \mathbf{s}(x))$  be the rank of label  $t$  in sorted list

$$\operatorname{rank}(t, \mathbf{s}(x)) = \sum_{y' \neq y} \mathbb{I}(s_{y'}(x) \geq s_y(x))$$

- ▶ For a single correct  $t$ , we can minimize the above rank

## Embedding type labels with WARP loss [6] (3)

- ▶ For multiple correct  $ts$ , there are various options to combine their ranks, e.g., sum
- ▶ For instance  $x_i, \mathbf{y}_i$ , consider **good** type  $t \in \mathbf{y}_i$ , **bad** type  $t' \notin \mathbf{y}_i$
- ▶ RANKSVM loss for such a pair would be  
$$\max\{0, 1 + s_{t'}(x) - s_t(x)\}$$
- ▶ To incorporate the rank signal of  $t$ , define overall WARP loss

$$\sum_{t \in \mathbf{y}_i} \sum_{t' \notin \mathbf{y}_i} \mathcal{R}(\text{rank}(t, \mathbf{s}(x))) \max\{0, 1 + s_{t'}(x) - s_t(x)\}$$

- ▶ Here  $\mathcal{R}$  transforms rank into weight; for precision at  $k$ , we can use  $\mathcal{R} = \sum_{1 \leq i \leq k} 1/i$
- ▶ Not convex

## Kernel WSABIE

- ▶ Earlier,  $s_t(x) = (Ax) \cdot (B\delta_t) = x^\top (A^\top B) \delta_t$
- ▶ Where  $Ax \in \mathbb{R}^H$  and  $B\delta_t \in \mathbb{R}^H$
- ▶  $A$  and  $B$  appear in only the form  $A^\top B \in \mathbb{R}^{D \times T}$ , but it is constrained to have rank at most  $H$  as a form of regularization
- ▶ Despite this, observed noisy “fill” in this matrix while training
- ▶ Let  $P \circ Q$  be the elementwise product of two matrices, i.e.,  $(P \circ Q)[d, t] = P[d, t] Q[d, t]$
- ▶ Google system uses  $K \in \{0, 1\}^{D \times T}$  as a feature selection or additional noise reduction mechanism

$$s_t(x) = x^\top (K \circ (A^\top B)) \delta_t$$

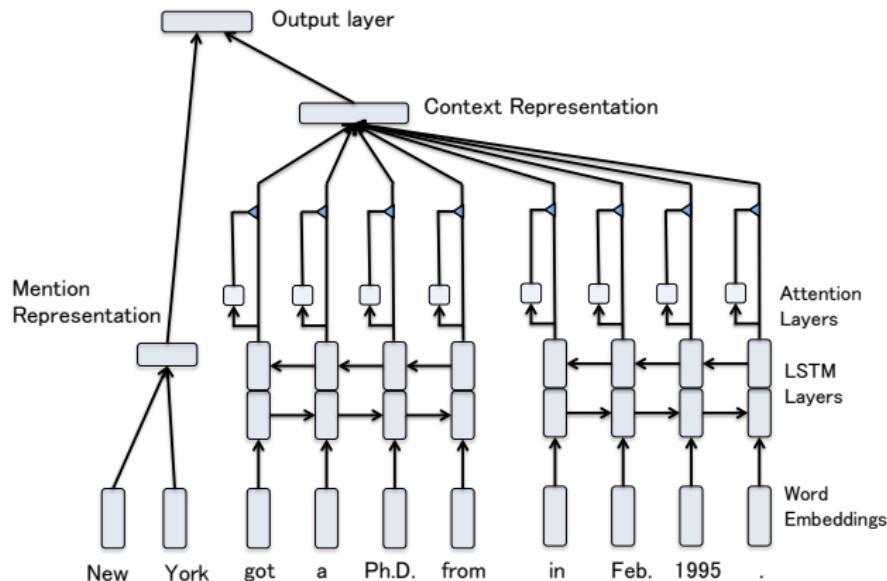
- ▶ If  $A[:, d]$  is among the 200 nearest neighbors of  $B[:, t]$ , set  $K[d, t] = 1$ , and 0 otherwise
- ▶  $K$  updated after every iteration (mini-batch?)

## Google fine-type system #2 performance

Method	P	R	F1
Ling and Weld (2012)	–	–	69.30
WSABIE	81.85	63.75	71.68
K-WSABIE	<b>82.23</b>	<b>64.55</b>	<b>72.35</b>

Table 4: Precision (P), Recall (R), and F1-score on the FIGER dataset for three competing models. We took the F1 score from Ling and Weld’s best result (no precision and recall numbers were reported). The improvements for WSABIE and K-WSABIE over the baseline are statistically significant ( $p < 0.01$ ).

## Bi-LSTM fine-type tagger [7]



She got a Ph.D from **New York** in Feb. 1995.

- ▶ Bi-LSTM on left and right context
- ▶ Average of word vectors of mention
- ▶ +Attention

## Bi-LSTM fine-type tagger details

- ▶ Let mention words be  $M = \{m\}$  with corresponding pretrained (focus) word vectors  $u(m)$  from word2vec or GloVe
- ▶ **Mention vector** is designed as  $v_m = (1/|M|) \sum_{m \in M} u(m)$
- ▶ Suppose we take  $C$  words of context from left and right
- ▶ Rightmost state from left context LSTM is  $\vec{h}_C^\ell$
- ▶ Leftmost state from right context LSTM is  $\overleftarrow{h}_1^r$
- ▶ **Context vector** is designed as  $v_c = \begin{bmatrix} \vec{h}_C^\ell \\ \overleftarrow{h}_1^r \end{bmatrix}$
- ▶ Each type  $t$  is predicted with

$$\Pr(t|\text{mention, context}) = \sigma \left( W_t \begin{bmatrix} v_m \\ v_c \end{bmatrix} \right)$$

# Computing $v_c$ with attention

Sentence	Prediction
... The film is a remake of [Secrets (1924)], a silent film starring Norma Talmadge ...	/film 0.986 /art 0.982
The film is a remake of Secrets (1924), a silent film starring [Norma Talmadge].	/person 0.999 /actor 0.987
... The festival brought together the foremost filmmakers, including Francois Truffaut, [Roman Polanski], Robert Enrico, and others.	/person 1.00 /director 0.963 /author 0.958 /artist 0.950 /actor 0.871
... Jim Hodges, the Democratic nominee, handily defeated Republican Governor [David Beasley] to become the 114th governor of South Carolina	/person 1.00 /politician 0.983
She is best known for roles in various TV Dramas and tokusatsu shows such as [Ultraseven X] and Kamen Rider Kiva.	/broadcasts_program 0.892

$$e_i^\ell = \tanh \left( W_e \begin{bmatrix} \vec{h}_i^\ell \\ \vec{h}_i^r \end{bmatrix} \right) \quad \text{L-R \& R-L states from left context}$$

$$e_i^r = \dots \quad \text{L-R \& R-L states from right context}$$

$$\tilde{a}_i^\ell = \exp \left( W_a e_i^\ell \right) \quad \text{Attend to important context words}$$

$$\tilde{a}_i^r = \dots$$

$$a_i^\ell = \frac{\tilde{a}_i^\ell}{\sum_{i=1}^C (\tilde{a}_i^\ell + \tilde{a}_i^r)} \quad \text{Normalize attention over left context}$$

$$v_c = \sum_{i=1}^C a_i^\ell \begin{bmatrix} \vec{h}_i^\ell \\ \vec{h}_i^r \end{bmatrix} + a_i^r \begin{bmatrix} \vec{h}_i^r \\ \vec{h}_i^\ell \end{bmatrix} \quad \text{Redefined context representation}$$

## LSTM and attention results

Models	P	R	F1
Ling and Weld (2012)	-	-	69.30
Yogatama et al. (2015)	<b>82.23</b>	64.55	72.35
Averaging Encoder	68.63	69.07	68.65
LSTM Encoder	72.32	70.36	71.34
Attentive Encoder	73.63	<b>76.29</b>	<b>74.94</b>

**Table 1:** Loose Micro Precision (P), Recall (R), and F1-score on the test set

Models	Strict	Loose Macro	Loose Micro
Ling and Weld (2012)	52.30	69.90	69.30
Yogatama et al. (2015)	-	-	72.25
Averaging Encoder	51.89	72.24	68.65
LSTM Encoder	55.60	73.95	71.34
Attentive Encoder	<b>58.97</b>	<b>77.96</b>	<b>74.94</b>

**Table 2:** Strict, Loose Macro and Loose Micro F1-scores

## Reducing (type) label noise [8]

- ▶ Fine type training data in the form of spans directly gold-labeled with types is rare
- ▶ Wikipedia has millions of pages of text with gold mentions of entities
- ▶ Wikipedia, DBpedia, Freebase, WikiData, ... have type hierarchies from which we can get all types that contain an entity
- ▶ However, most of these types are not relevant at any given mention of the entity
- ▶ Training all these types using this textual context would pollute the type models
- ▶ Notation: entity  $e$ , with mention contexts  $C_e = \{c_{ei}\}$  (if  $e$  is understood, will drop it)
- ▶  $e$  is a member of types in  $T_e$ , specified by KG

## Reducing (type) label noise [8] (2)

- ▶ I.e., each  $e$  associated with  $\mathbf{y}_e \in \{0, 1\}^{|\mathcal{T}|}$ , a few-hot vector of types
- ▶ Less realistic to assume per-context gold labels (except to eval fine-type system)
- ▶ Each mention context is an **instance**
- ▶ I.e., each entity is associated with multiple instances
- ▶ In general each entity has multiple valid **labels** (types)
- ▶ Therefore, a multi-instance multi-label (MIML) setting
- ▶ Each context associate with ~~~~~ (one/more) types?

## MIML approach to fine typing

- ▶ Each context  $c_i$  will be represented by a fixed-size vector  $\mathbf{c}_i \in \mathbb{R}^H$  (defined later)
- ▶ A first-cut per-mention predictor is a logistic regression:  
$$\Pr(t|c_i) = \sigma(\mathbf{w}_t \cdot \mathbf{c}_i + b_t)$$
- ▶ Note multiple  $t$  can have score close to 1
- ▶ Next, we aggregate in various ways over contexts
- ▶ **MIML-MAX**: Each type  $t \in T_e$  is supported by one best context:  
$$\Pr(t|e) = \max_{c \in C_e} \Pr(t|c)$$
- ▶ Ignores all smaller endorsements
- ▶ **MIML-AVG**:  
$$\Pr(t|e) = \frac{1}{|C_e|} \sum_{c \in C_e} \Pr(t|c)$$
- ▶ Binary cross entropy  $\text{BCE}(y, y') = -y \log y' - (1 - y) \log(1 - y')$
- ▶ All  $\mathbf{w}_t$ s can be trained using cross-entropy loss  
$$L(\{\mathbf{w}_t\}) = \sum_e \sum_t \text{BCE}(y_{et}, \Pr(t|e; \mathbf{w}_t))$$

## MIML approach to fine typing (2)

- ▶ MIML-ATT: Aggregate with attention over contexts
- ▶ Apart from  $\mathbf{w}_t$ , associate each  $t$  with another vector  $\mathbf{v}_t$
- ▶ Mention contexts of entity  $e$  compete for attention:
$$\alpha_{i,t} = \frac{\exp(\mathbf{c}_i \cdot \mathbf{v}_t)}{\sum_{i'} \exp(\mathbf{c}_{i'} \cdot \mathbf{v}_t)}$$
- ▶ Now we build an attention-weighted context representation:
$$\mathbf{a}_t = \sum_i \alpha_{i,t} \mathbf{c}_i$$
- ▶ Use  $\mathbf{a}_t$  in place of  $\mathbf{c}_i$  before:  $\Pr(t|e) = \sigma(\mathbf{w}_t \cdot \mathbf{a}_t + b_t)$
- ▶ Loss as before
- ▶ Additional “deepness”:  $\alpha_{i,t} = \frac{\exp(\mathbf{c}_i^\top \mathbf{M} \mathbf{v}_t)}{\sum_{i'} \exp(\mathbf{c}_{i'}^\top \mathbf{M} \mathbf{v}_t)}$ , where  $\mathbf{M}$  measures the similarity between context and  $\mathbf{v}_t$

## Context representation $c_i$ using convnet

- ▶ At the input, read word embeddings
- ▶ Apply narrow convnets separately to left and right context of mention to get  $\phi_\ell(c), \phi_r(c)$
- ▶ Concatenate into  $\phi(c)$  and compute  $c = \tanh(\mathbf{S}\phi(c))$  where  $\mathbf{S}$  is more model weights
- ▶ So overall we have these model weights:
  - ▶ Global  $\mathbf{M}, \mathbf{S}$
  - ▶ Global weights in convnet  $\phi$
  - ▶  $\mathbf{w}_t, \mathbf{v}_t, b_t$  for each type
  - ▶ Word embeddings (if fine tuned after pretraining)
- ▶ Between  $\mathbf{w}_t, \mathbf{v}_t$ , is there a usable/interpretable representation of type  $t$ ?
- ▶ (How) do they relate to entity embeddings as in ent2vec?

## Noise mitigation results

	$P@1$ all	$F_1$ all	$F_1$ head	$F_1$ tail	MAP
1 MLP	74.3	69.1	74.8	52.5	42.1
2 MLP+MIML-MAX	74.7	59.2	50.7	46.8	41.3
3 MLP+MIML-AVG	77.2	70.6	74.9	56.2	45.0
4 MLP+MIML-MAX-AVG	75.2	71.2	76.4	56.0	47.1
5 MLP+MIML-ATT	81.0	72.0	76.9	59.1	48.8
6 CNN	78.4	72.2	77.3	56.3	47.6
7 CNN+MIML-MAX	78.6	62.2	53.5	49.7	46.6
8 CNN+MIML-AVG	80.8	73.5	77.7	59.2	50.4
9 CNN+MIML-MAX-AVG	79.9	74.3	79.2	59.8	53.3
10 CNN+MIML-ATT	83.4	75.1	79.4	62.2	55.2
11 EntEmb	80.8	73.3	79.9	57.4	56.6
12 FIGMENT	81.6	74.3	80.3	60.1	57.0
13 CNN+MIML-ATT+EntEmb	<b>85.4</b>	<b>78.2</b>	<b>83.3</b>	<b>66.2</b>	<b>64.8</b>

- ▶ ClueWeb with FACC1 entity annotations
- ▶ Freebase entities mapped to 102 FIGER types
- ▶ 4.3 million contexts
- ▶ Head means > 100, tail < 5 mentions

## Noise mitigation summary

- ▶ **Corpus-oriented** data set has gold type/s for each entity mention
- ▶ Number of mentions of  $e$  with a gold type  $t$  is indication of how strongly  $e \in t$
- ▶ **KG-oriented** data set (no corpus mentions) has gold type set  $T_e$  for each entity
- ▶ Assumed incomplete; task is to discover new  $e \in t$
- ▶ A case of “knowledge base completion” or **KBC**
- ▶ In KG+corpus data sets, each corpus mention of  $e$  supports a (small, possibly empty) subset of  $T_e$
- ▶ For  $t$  to be included in  $T_e$ , at least one mention must support it
- ▶ Evaluation can be per-mention (costly) or KBC (held-out entities or  $e \in t$  instances)
- ▶ Caution needed designing folds for latter case

## References

- [1] X. Ling and D. S. Weld, "Fine-grained entity recognition." in *AAAI*, 2012. [Online]. Available: <http://xiaoling.github.io/pubs/ling-AAAI12.pdf>
- [2] D. Gillick, N. Lazic, K. Ganchev, J. Kirchner, and D. Huynh, "Context-dependent fine-grained entity type tagging," *arXiv preprint arXiv:1412.1820*, 2014. [Online]. Available: <https://arxiv.org/pdf/1412.1820.pdf>
- [3] S. Murty, P. Verga, L. Vilnis, and A. McCallum, "Finer grained entity typing with TypeNet," *arXiv preprint arXiv:1711.05795*, 2017. [Online]. Available: <https://arxiv.org/pdf/1711.05795.pdf>
- [4] S. Chakrabarti, S. Kasturi, B. Balakrishnan, G. Ramakrishnan, and R. Saraf, "Compressed data structures for annotated web search," in *WWW Conference*, 2012, pp. 121–130. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/www2012/>

## References (2)

- [5] S. Yavuz, I. Gur, Y. Su, M. Srivatsa, and X. Yan, "Improving semantic parsing via answer type inference." in *EMNLP Conference*, 2016, pp. 149–159. [Online]. Available: [http://cs.ucsb.edu/~ysu/papers/emnlp16\\_type.pdf](http://cs.ucsb.edu/~ysu/papers/emnlp16_type.pdf)
- [6] D. Yogatama, D. Gillick, and N. Lazic, "Embedding methods for fine grained entity type classification," in *ACL Conference*, 2015, pp. 26–31. [Online]. Available: <http://anthology.aclweb.org/P/P15/P15-2048.pdf>
- [7] S. Shimaoka, P. Stenetorp, K. Inui, and S. Riedel, "An attentive neural architecture for fine-grained entity type classification," *arXiv preprint arXiv:1604.05525*, 2016. [Online]. Available: <https://arxiv.org/pdf/1604.05525.pdf>
- [8] Y. Yaghoobzadeh, H. Adel, and H. Schütze, "Noise mitigation for neural entity typing and relation extraction," *arXiv preprint arXiv:1612.07495*, 2016. [Online]. Available: <https://arxiv.org/pdf/1612.07495.pdf>

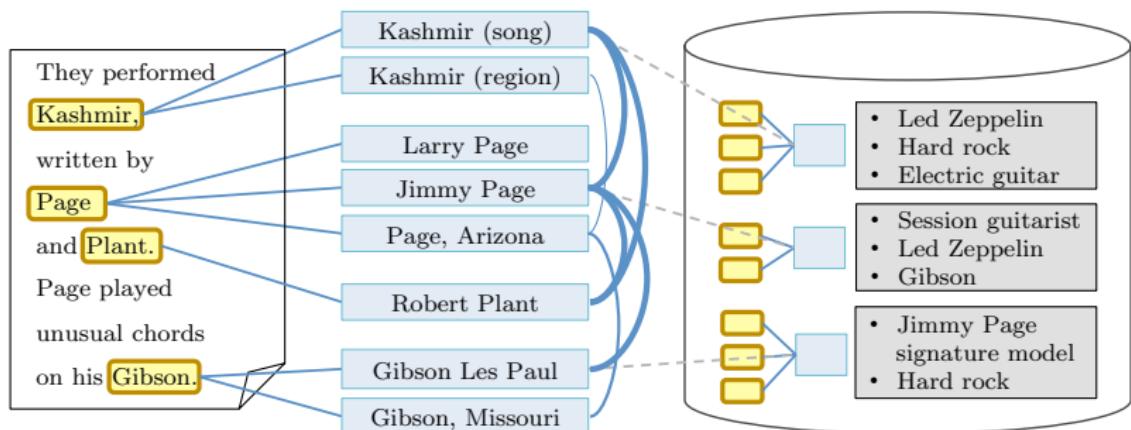
## Named entity disambiguation

## Entity linking — working definition and motivation

- ▶ From coarse NER to fine types to specific entities with canonical IDs in knowledge graph (KG), e.g.,  
[http://en.wikipedia.org/wiki/Michael\\_Jordan](http://en.wikipedia.org/wiki/Michael_Jordan) or  
[http://en.wikipedia.org/wiki/Michael\\_I.\\_Jordan](http://en.wikipedia.org/wiki/Michael_I._Jordan)
- ▶ Many choices of KGs: Wikipedia, WikiData, Freebase, Google KG, Bing Satori, ...
- ▶ An entity can have many **aliases**: Michael Jordan, Mike, Jordan
- ▶ Conversely, *Jordan* can refer to river, country, and lots of people
- ▶ A passage may **mention**<sup>5</sup> an entity; around the mention  $m$  is a **context**  $c$  from which we can observe **context features**
- ▶ If the mention string matches an alias of an entity  $e$ , the entity becomes a **candidate**
- ▶  $\Gamma(m)$  is the set of all candidates of mention  $m$

## Entity linking — working definition and motivation (2)

- ▶ For each mention, the goal is to choose one or zero (out-of-KG, reject, null,  $\perp$ , NA) candidate
- ▶ Each mention is a multiclass, single-label classification problem, but they are inter-related



- ▶ Entity label at mention  $m_i$  is  $y_i$ ; gold label is  $y_i^*$

## Entity linking — working definition and motivation (3)

- ▶ Motivation: complex query responses involving joins
  - ▶ Company  $?c$  in Korea makes phone  $?p$  under \$400 with OLED display — instantiate all possible  $\langle ?c, ?p \rangle$
  - ▶ Need to recognize  $?c$ ,  $?p$  as (single) company and phone in different contexts provide evidence for subqueries

---

<sup>5</sup>Detecting mention boundaries is difficult [1] but is outside our scope.

## Some distinctions from WSD

- ▶ Word sense disambiguation (WSD) is largely about common words, not references to specific entities
  - ▶ 42 senses of “run” in WordNet
  - ▶ Part of speech helps a fair bit
  - ▶ Identifying mention boundary is easy
- ▶ Entity catalog typically richer info source than dictionary
  - ▶ Broader category system
  - ▶ Part of speech is largely “proper noun” and not as helpful
- ▶ Entity disambiguation goals:
  - ▶ Identify that a sequence of tokens is a potential mention
  - ▶ Capture suitable context around to form spot  $s$
  - ▶ Assign  $s$  to a suitable entity  $\gamma$  in catalog
  - ▶ Or claim that there is no suitable  $\gamma$

## Why annotate?

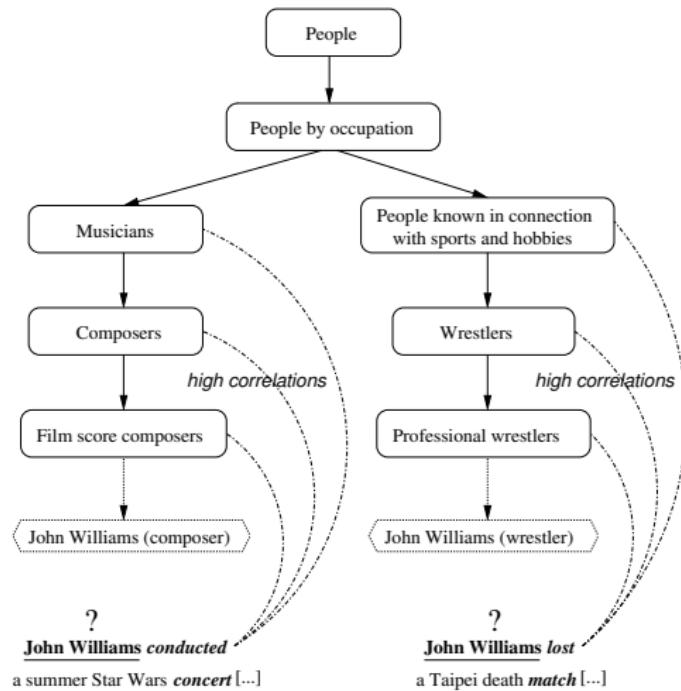
- ▶ Make raw text look like Wikipedia with definitional and informational links (most systems)
  - ▶ Annotate first occurrence only
  - ▶ Annotate only on-topic entities
  - ▶ Use discretion to avoid “hyperlink fatigue”
- ▶ **Index** the annotations to enable advanced search (our focus)
  - ▶ Exhaustive annotation
  - ▶ Make no whole-document topic judgment

## More about catalog representation

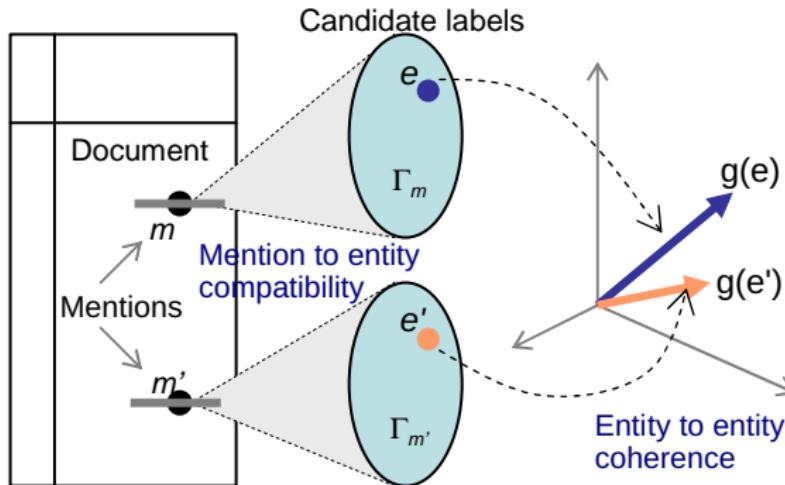
- ▶ Pattern after WordNet, Wikipedia, Freebase, ...
- ▶ Each entity  $\gamma$  has associated **description** or **definition** page
- ▶ Descriptions **link** to other related entities  $\gamma'$
- ▶ Entities belong to one or more **categories**
- ▶ Categories (physicist) are **subcategories** of others (scientist)
- ▶ Links may be “incidental”
- ▶ Categories and super-categories may be noisy: *Machine learning researcher* more meaningful than *Living people* or *Year of birth missing*
- ▶ Cycles in is-a “hierarchy” ?

# Local signals to choose $e$ from $\Gamma(m)$

- ▶ Match between context and entity
- ▶ Entity representations
  - ▶ Text on definition pages in Wikipedia
  - ▶ Text from gold mention contexts
  - ▶ Types that contain the entity
- ▶ Between context and types containing entity [2]
- ▶ Between page topic/s and entity type/s [3]



## Integrating local and global signals [4, 3]



- ▶ Some entity pairs are more **coherent** than others
- ▶ Coherence may be measured in different ways
- ▶ Choose per-mention entity labels to maximize pairwise coherence as well as local compatibility
- ▶ Intractable in general; heuristic approximations common

## SemTag [5]

- ▶ Used Stanford TAP ontology (72,000 entities)
- ▶ Set of classes  $C$ , subclass relation  $S \subseteq C \times C$ , set of instances (entities)  $I$ , many-to-many type relation  $T \subseteq I \times C$
- ▶  $i$  has class  $c_1$  and  $c_1$  subclass of  $c_2$  implies  $i$  has class  $c_2$
- ▶ Entity taxonomy is a DAG,  $\pi(v)$  is the path up from  $v$  to root node  $r$
- ▶ Taxonomy node  $v$  has label set  $L(v)$ , e.g., nodes corresponding to cats, football, computers and cars all contain the label ‘jaguar’

## SemTag output example

The <resource ref="http://tap.stanford.edu/BasketballTeam\_Bulls">Chicago Bulls</resource> announced yesterday that <resource ref="http://tap.stanford.edu/AthleteJordan, Michael">Michael Jordan</resource> will ...

- ▶ Functionally identical to inserting Wikipedia links in free-form text
- ▶ Wikipedia is more organic than TAP; has poorer quality category hierarchy

## SemTag disambiguation

- ▶  $\text{sim}(u, s) \in [0, 1]$  is a local similarity between catalog node  $u$  and (context of) spot  $s$
- ▶  $\text{sim}(\cdot, \cdot) = \frac{1}{2}$  is “most uncertain”
- ▶ Node  $v$  is **eligible** for spot  $s$  if

$$\text{root } r \neq \arg \max_{u \in \pi(v)} \text{sim}(u, s)$$

i.e., some node on  $\pi(v)$  other than root most similar to  $s$

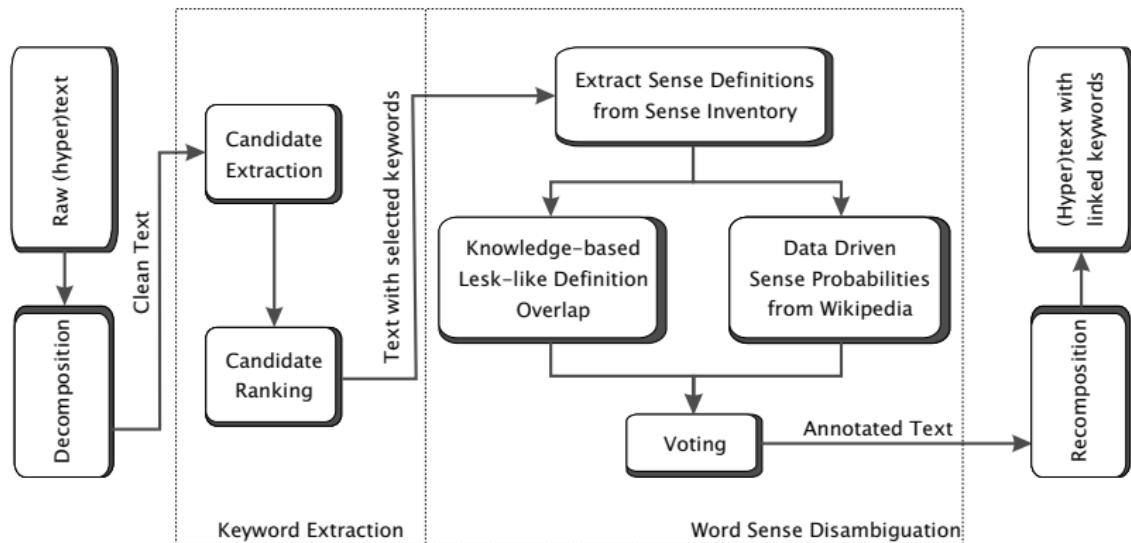
- ▶ Supplement eligibility with human-judged scores of **reliability** at each node  $u$ 
  - ▶  $m_u^a$  = probability that spots for subtree rooted at  $u$  are “on topic”
  - ▶  $m_u^s$  = probability that automatic eligibility judgment is correct

## SemTag TBD algorithm

- ▶ To decide whether to link spot  $s$  to node  $v$  ...
- ▶ Find nearest ancestor  $u$  of  $v$  that has human-judged reliability scores
- ▶ If  $|\frac{1}{2} - m_u^a| > |\frac{1}{2} - m_u^s|$ , return  $\text{sign}(m_u^a - \frac{1}{2})$
- ▶ Else if  $m_u^s > \frac{1}{2}$  (eligibility judgment is often correct), return  $\text{eligible}(c, u)$
- ▶ Else (eligibility judgment is often wrong) return  $1 - \text{eligible}(c, u)$

(Can regard as a simple hand-tuned form of stacked learning)

# Wikify! [6]



- ▶ Two-phase process
- ▶ First identify token spans “worthy of annotation”
- ▶ Then choose entity labels

## Sample annotations

In 1834, Sumner was admitted to the [[bar (law)|bar]] at the age of twenty-thre, and entered private practice in Boston.

---

It is danced in 3/4 time (like most waltzes), with the couple turning approx. 180 degrees every [[bar (music)|bar]].

Vehicles of this type may contain expensive audio players, televisions, video players, and [[bar (counter)|bar]]s, often with refrigerators.

---

Jenga is a popular beer in the [[bar (establishment)|bar]]s of Thailand

---

This is a disturbance on the water surface of a river or estuary, often cause by the presence of a [[bar (landform)|bar]] or dune on the riverbed.

## Choosing token spans to annotate (“spotting”)

- ▶ Wikify! follows the Wikipedia philosophy
- ▶ Use some score to rank candidate spans
- ▶ TFIDF of a token in a document

$\chi^2$ test:	count of token in doc	count of all other tokens in doc
	count of token in other docs	count of all other tokens in other docs

- ▶ “Keyphraseness” — In how many Wikipedia documents is the same term made a link anchor?
- ▶ (They only consider as candidates words which appear at least five times in Wikipedia)

## Disambiguation

Wikify! compares two local techniques:

- ▶ “Knowledge-based approach” — similarity between Wikipedia page text of entity  $\gamma$  and context words in spot  $s$
- ▶ “Data-driven approach” — similarity between context of known links to  $\gamma$  and context words in spot  $s$
- ▶ “Context” consists of  $\pm 3$  words around mention, their parts of speech, salient words chosen from whole document

## Results

- ▶ “Data-driven” better than “knowledge-based”
- ▶ Consensus (agreement) has highest precision

Method	Words		Evaluation		
	(A)	(C)	(P)	(R)	(F)
Baselines					
Random baseline	6,517	4,161	63.84	56.90	60.17
Most frequent sense	6,517	5,672	87.03	77.57	82.02
Word sense disambiguation methods					
Knowledge-based	6,517	5,255	80.63	71.86	75.99
Feature-based learning	6,517	6,055	92.91	<b>83.10</b>	<b>87.73</b>
Combined	5,433	5,125	<b>94.33</b>	70.51	80.69

Upload a text or HTML file:

Browse...

Or type a URL:

[http://news.bbc.co.uk/2/hi/south\\_asia/539](http://news.bbc.co.uk/2/hi/south_asia/539)Density  
thin → thickLink color:  
white  
blue  
red

Wikify!

Wikifier

HOME | HELP | ABOUT | CONTACT

• UK version • International version | About the versions



BBC News in video and audio

News Front Page



Africa

Americas

Asia-Pacific

Europe

Middle East

South Asia

UK

Business

Health

Science/Nature

Technology

Entertainment

Have Your Say

In Pictures

Country Profiles

Special Reports

Programmes

RELATED BBC SITES

Done

Last Updated: Thursday, 28 September 2006, 19:20 GMT 20

E-mail this to a friend

Printable version

## Nato to extend Afghan operations

Nato has announced that it will extend its mission in Afghanistan to cover the whole of the insurgency-hit country.

The move will take the alliance into the eastern parts of Afghanistan and bring up to 12,000 American troops under Nato command.

A Nato official said the decision would be implemented in the next few weeks.

The announcement came as the US military said that militant attacks near the Pakistani border had tripled in some areas.

The rise in activity comes despite a peace agreement meant to end violence by pro-Taliban militants in Pakistan's North Waziristan border area.



Nato will now command more than 30,000 troops in Afghanistan

## Afghanistan

From Wikipedia, the free encyclopedia

This article is best suited for Wikify!  
Please consider transferring content to  
[Wikipezia](#). See [Wikipezia:Long article layout and W&H](#).

**Afghanistan** (officially the **Islamic Republic of Afghanistan**; Persian (Dari): جمهوری اسلامی افغانستان; Pashto: د افغانستان اسلامی جمیوونت) is a landlocked country at the crossroads of Asia and the Middle East. Generally considered a part of Central Asia, it is sometimes ascribed to a regional bloc in either the Middle East or South Asia, as it has cultural



S FUTURE  
In Depth  
Part?  
stan

## Military of the United States

From Wikipedia, the free encyclopedia

The military of the United States, officially known as the United States Armed Forces, consist of the:

- United States Army
- United States Marine Corps
- United States Navy
- United States Air Force
- United States Coast Guard

All the services are under the command of the President of the United States. All of the services except the Coast Guard are part of the Department of Defense, which is controlled by the Secretary of Defense. In peacetime the Coast Guard is part of the

► OPEN Afghanistan at-a-glance

## Modeling local compatibility

- ▶ Feature vector  $f_s(\gamma) \in \mathbb{R}^d$  expresses local textual compatibility between (context of) spot  $s$  and candidate label  $\gamma$
- ▶ One element of  $f_s(\gamma)$  might be the TFIDF cosine similarity between tokens from the context of spot  $s$  (say  $\pm 10$  tokens) and whole page of description for entity  $\gamma$
- ▶ Another element may be derived of “anchor text” match:
  - ▶ Find all links to  $\gamma$  from within Wikipedia
  - ▶ Collect anchor text from all these links in a bag of words
  - ▶ Find TFIDF cosine similarity between this bag and the spot context  $s$

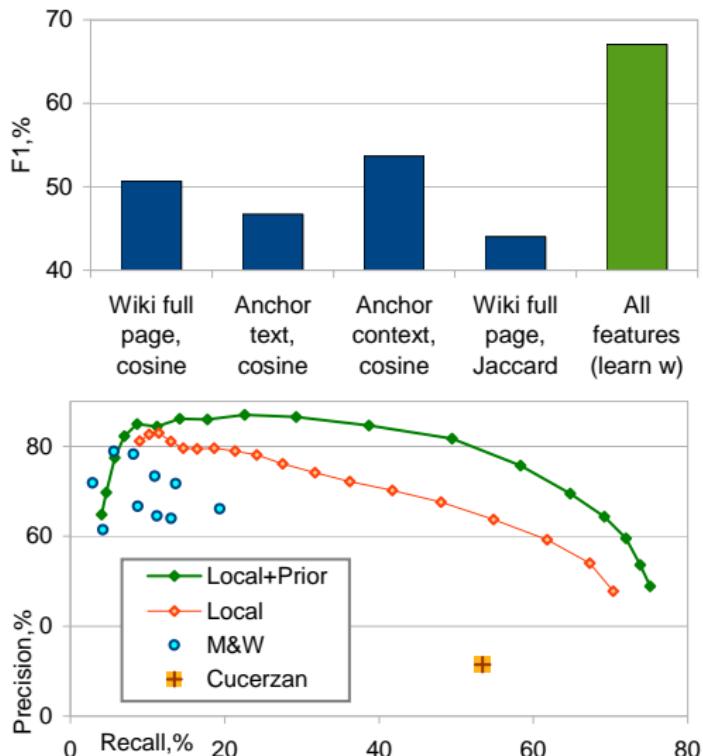
## The sense probability prior

- ▶ What entity does “Intel” refer to?
  - ▶ Chip design and manufacturing company
  - ▶ Fictional cartel in a 1961 BBC TV serial
- ▶  $\Pr_0(\gamma|s)$  is very high for chip maker, low for cartel
- ▶ Append element  $\log \Pr_0(\gamma|s)$  to  $f_s(\gamma)$
- ▶ “log” will be explained later

## Node score

- ▶ Node scoring **model**  $w \in \mathbb{R}^d$
- ▶ Node score defined as  $w^\top f_s(\gamma)$
- ▶  $w$  is trained to give suitable relative weights to different compatibility measures and aggregate the evidence
- ▶ During test time, **greedy** choice local to  $s$  would be  $\arg \max_{\gamma \in \Gamma_s} w^\top f_s(\gamma)$
- ▶ Early algorithms are variations on this theme

## Effect of learning single-mention scores

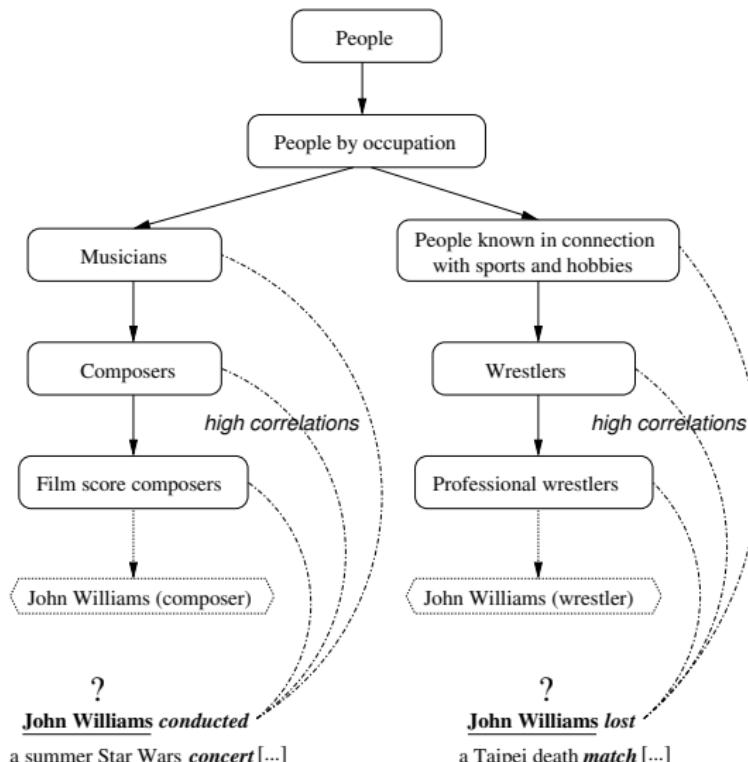


- ▶ Learning  $w$  is better than commonly-used single features
- ▶ Enough to beat some collective approaches (soon)

## Limitations of $\text{sim}(\gamma, s)$

- ▶ Training data is sparse
- ▶ Direct overlap of words between description of entity  $\gamma$  and context of spot  $s$  may be limited
- ▶ But overlap between **ancestors** of  $\gamma$  and context of  $s$  may be more reliable

# Word-category correlations



## Designing tree kernels

- ▶ Let  $C(\gamma)$  be all ancestor categories of entity  $\gamma$
- ▶ Let  $T(s)$  be the text in the context of spot  $s$
- ▶ For every word  $w$  and every all categories  $c$ , define a feature

$$\phi_{w,c}(s, \gamma) = \begin{cases} 1 & \text{if } w \in T(s) \text{ and } c \in C(\gamma) \\ 0 & \text{otherwise} \end{cases}$$

- ▶ Run through all possible  $w, c$ , e.g., (“conducted”, *musician*), (“concert”, *wrestler*)
- ▶ Pad  $(\phi_{w,c})$  with local compatibility features
- ▶ Finally, get feature vector  $\Phi(s, \gamma)$

## Learning

- ▶ Model as classification: correct/incorrect  $(s, \gamma)$  pair should be labeled  $+1/-1$  respectively
- ▶ Similar to sequence labeling:  $\arg \max_{\gamma} w^T \Phi(s, \gamma)$ ; same max-margin training
- ▶ What about spots that do not have any suitable entity in the catalog?
- ▶ Out-of-catalog entity  $\hat{\gamma}$ , with  $C(\hat{\gamma}) = \emptyset$  and  $T(\hat{\gamma}) = \emptyset$
- ▶ One last feature element  $\phi^\wedge(s, \gamma) = [\![\gamma = \hat{\gamma}]\!]$
- ▶ Equivalent to automatically learning a (lower) threshold on  $w^T \Phi(s, \gamma)$

## Tree kernel results

Data set	TreeKernel	TextOnly
People by occupation, top 110	0.772	0.615
Ditto, all 540	0.684	0.558
Ditto, categories with $\geq 20$ entities	0.680	0.554

- ▶ Summary: tree kernel better than comparing only text

## Modeling entity relatedness from catalog

- ▶ Some entity pairs are more compatible than others
- ▶ Better to choose per-mention entity labels to maximize pairwise compatibility
- ▶ Compatibility may have different notions
- ▶ Entities belong to related types, e.g., soccer coaches, clubs, players [4, 3]
- ▶ Frequently co-cited from Web/Wikipedia pages [7]
- ▶ Entities connected by short path in knowledge graph [8]
- ▶ (Similarity between vector embeddings of entities based on corpus mentions — soon)
- ▶ How related are two entities  $\gamma, \gamma'$  in Wikipedia?
- ▶ Embed  $\gamma$  in some space using  $g : \Gamma \rightarrow \mathbb{R}^c$
- ▶ Define relatedness  $r(\gamma, \gamma') = g(\gamma) \cdot g(\gamma')$  or related

## Modeling entity relatedness from catalog (2)

- ▶ Cucerzan's proposal:  $c = \text{number of categories}$ ;  $g(\gamma)[\tau] = 1$  if  $\gamma$  belongs to category  $\tau$ , 0 otherwise

$$r(\gamma, \gamma') = \frac{g(\gamma)^\top g(\gamma')}{\sqrt{g(\gamma)^\top g(\gamma)} \sqrt{g(\gamma')^\top g(\gamma')}},$$

(standard cosine)

- ▶ Milne and Witten's proposal:  $c = \text{number of Wikipedia pages}$ ;  $g(\gamma)[p] = 1$  if page  $p$  links to page  $\gamma$ , 0 otherwise

$$r(\gamma, \gamma') = \frac{\log \frac{|g(\gamma) \cap g(\gamma')|}{|g(\gamma) \cup g(\gamma')|}}{\log \frac{c}{\min\{|g(\gamma)|, |g(\gamma')|\}}}$$

- ▶ Related to Jaccard
- ▶ With voice of small inlink sets attenuated
- ▶ Combination of above [9]

## A joint local+global objective

- ▶ Notation: mentions written variously as  $m_i, s_i$ ;  $s_i$  includes  $m_i$  and features from context  $c_i$
- ▶ Entity labels written variously as  $\gamma_i, y_i, e_i$
- ▶  $\Psi(e_i, m_i, c_i)$  is the local score of entity  $e_i$  for mention  $m_i$  with context  $c_i$
- ▶  $\Phi(e_i, e_j)$  is the pairwise coherence between the entities chosen for mentions  $i, j$
- ▶ For whole document, let  $\mathbf{e}, \mathbf{m}, \mathbf{c}$  be the sequence of  $n$  entity labels, mentions, and contexts
- ▶ Overall objective is to maximize wrt  $\mathbf{e}$

$$g(\mathbf{e}, \mathbf{m}, \mathbf{c}) = \underbrace{\frac{1}{n} \sum_i \Psi(e_i, m_i, c_i)}_{\text{local}} + \underbrace{\frac{1}{\binom{n}{2}} \sum_{i \neq j} \Phi(e_i, e_j)}_{\text{global}}$$

## A joint local+global objective (2)

- ▶ (Conditional) probabilistic graphical model with complete graph
- ▶ Aka the quadratic assignment problem
- ▶ Difficult NP-hard problem
- ▶ Heuristics: leave-one-out [10], easy-mention-first [7],  
hill-climbing [4, 11], LP relaxation [4], multifocal attention [12]

## Leave-one-out disambiguation [10]

- ▶ Let  $\Gamma_0 = \bigcup_i \Gamma(m_i)$  be all possible entity disambiguations for all mentions on a page
- ▶ Precompute the average entity representation vector  
$$g(\Gamma_0) = \sum_{\gamma \in \Gamma_0} g(\gamma)$$
- ▶ Score of candidate label  $\gamma$  for spot  $s$  depends on two factors multiplied together
- ▶ The local compatibility score as before
- ▶ 
$$g(\gamma)^\top g(\Gamma_0 \setminus \{\gamma\}) = g(\gamma)^\top \sum_{\gamma' \in \Gamma_0 \setminus \gamma} g(\gamma')$$
- ▶ Note that  $\Gamma_0 \setminus \gamma$  still contains contributions from entities that cannot be used simultaneously to label the page
- ▶  $g(\Gamma_0 \setminus \gamma)$  may not be a representative feature vector

# Commonness, usefulness, relatedness

## Depth-first search

From Wikipedia, the free encyclopedia

**Depth-first search (DFS)** is an algorithm for traversing or searching a tree structure or graph. One starts at the root (selecting some node as the root in the graph case) and explores as far as possible along each branch before backtracking.

Formally, DFS is an uninformed search that progresses by expanding the first child node of the search tree that appears and thus going deeper and deeper until a goal node is found, or until it hits a node that has no children. Then the search backtracks, returning to the most recent node it hadn't finished exploring. In a non-recursive implementation, all freshly expanded nodes are added to a LIFO stack for exploration.

sense	commonness	relatedness
Tree	92.82%	15.97%
Tree (graph theory)	2.94%	59.91%
<b>Tree (data structure)</b>	<b>2.57%</b>	<b>63.26%</b>
Tree (set theory)	0.15%	34.04%
Phylogenetic tree	0.07%	20.33%
Christmas tree	0.07%	0.0%
Binary tree	0.04%	62.43%
Family tree	0.04%	16.31%
...		

- ▶ “Tree” has many senses, common and rare
- ▶ But a low probability sense may be the correct one, based on relatedness to unambiguous anchor entities mentioned near “tree”
- ▶ Not all anchors equally useful: “until” vs. “LIFO”

## Milne and Witten's recipe

- ▶ Identify unambiguous spots  $S_!$  from all spots  $S_0$
- ▶ Denote  $\Gamma_! = \bigcup_{s \in S_!} \Gamma_s$ , note that  $\Gamma_! \xleftrightarrow{1:1} S_!$
- ▶ Ambiguous spot  $s \mapsto \Gamma_s$ , have to pick  $\gamma \in \Gamma_s$
- ▶ Each candidate  $\gamma$  is scored based on three signals
  - Commonness of  $\gamma$ , i.e., sense probability prior  $\Pr_0(\gamma|s)$
  - Average relatedness to anchor entities  $\gamma_!$ , weighted by the usefulness  $u(\gamma_!)$  of  $\gamma_!$

$$\frac{\sum_{\gamma_! \in \Gamma_! \setminus \gamma} u(\gamma_!) r(\gamma, \gamma_!)}{\sum_{\gamma_! \in \Gamma_! \setminus \gamma} u(\gamma_!)}$$

$$\text{where } u(\gamma) = \sum_{\gamma'' \in \Gamma_! \setminus \gamma'} r(\gamma', \gamma'')$$

Overall context quality for the spot,  $\sum_{\gamma_!} u(\gamma_!)$

## Milne and Witten's recipe (2)

- ▶ These three signals are presented as features to a classifier  
(bagged decision tree worked best)
- ▶ The label is whether  $\gamma$  is correct for  $s$

## M&W results

	recall	precision	f-measure
Random sense	56.4	50.2	53.1
Most common sense	92.2	89.3	90.7
Medelyan <i>et al.</i> (2008)	92.3	93.3	92.9
Most valid sense	95.7	<b>98.4</b>	<b>97.1</b>
All valid senses	<b>96.6</b>	97.0	96.8

- ▶ Random sense gives precision over  $\frac{1}{2}$ , only around two senses per spot
- ▶ Recall is as per (reticent) Wikipedia annotation policy

correct	76.4
incorrect (wrong destination)	0.9
incorrect (irrelevant and/or unhelpful)	19.8
incorrect (unknown reason)	2.9

## Hill-climbing [11]

- ▶ Two stages, **ranker** followed by **linker**
- ▶ Ranker obtains best non-null label for each mention
- ▶ Linker decides whether to replace best label with NA

**for** each mention  $m_i$  **do**

    construct disambiguation candidates  $\Gamma_i$

    run **ranker** to get best non-null disambiguation  $y_i$

**for** mentions  $m_i$  in some arbitrary order **do**

**if** changing  $y_i$  to null improves collective objective **then**  
        commit to change

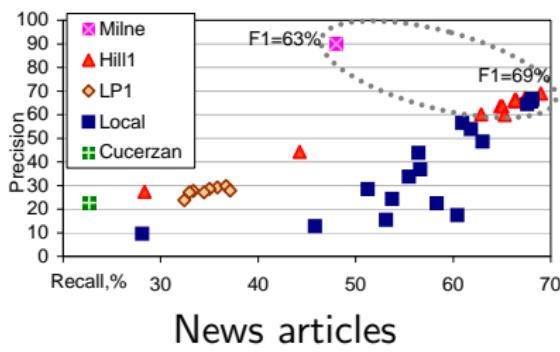
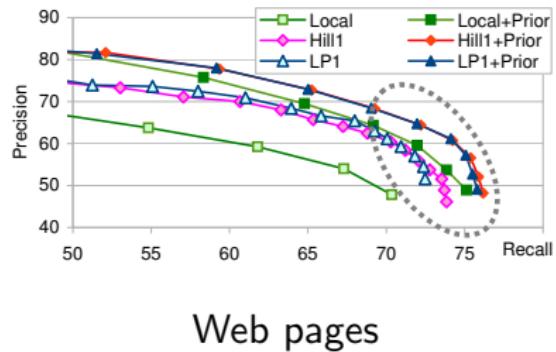
More details

## Integer program

- ▶ Let  $i$  index mentions and  $e$  index candidate entities
- ▶ Decision variable  $z_{ie} \in \{0, 1\}$  is 1 if mention  $i$  gets label  $e$  and 0 otherwise
- ▶ For each mention  $i$ ,  $\sum_{e \in \Gamma_i} z_{ie} \leq 1$  (zero or one label per mention from among candidates)
- ▶ Local node log-potential for mention  $i$  is  $\phi_i(e)$
- ▶ Local objective is  $\sum_i \sum_e \phi_i(e) z_{ie}$
- ▶ Auxiliary decision variables  $p_{i,e,i',e'} \in \{0, 1\}$  for all mention and label pairs
- ▶ Constraints for all  $i, e, i', e'$ ,  $p_{i,e,i',e'} \leq z_{ie}$  and  $p_{i,e,i',e'} \leq z_{i'e'}$
- ▶ Global objective is  $\sum_{i,e,i',e'} p_{i,e,i',e'} \psi_{ii'}(e, e')$
- ▶ Relax to  $z_{i,e}, p_{i,e,i',e'} \in [0, 1]$  (not a nice relaxation, cannot round to provably good discrete solutions)

# Benefits of collective labeling

- ▶ Two different data sets (Web, newswire)
- ▶ Can significantly push recall while preserving precision
- ▶ Improves upon Milne&Witten [7], Cucerzan [10]



## Multifocal attention [12]

- ▶ Consider again the **all-pairs** global term  $\sum_{i \neq j} \Phi(e_i, e_j)$
- ▶ Entities in doc may not all be in one type cluster; e.g.,  $e_i$  may be a politician and  $e_j$  a real-estate baron
- ▶ KG may not know of common type-to-type relations, e.g., cricketers and business tycoons, or politicians and real estate barons
- ▶ Less salient entity  $e_i$  may not find enough  $\Phi$  support from all other entities  $e_j$
- ▶ Asserting all-pairs potentials across coherent clusters needlessly adds noise floor to objective
- ▶ Discussed by Kulkarni et al. [4] but not addressed

## Single link baseline

- ▶ As an extreme simplification of the clique potential, for each mention, find **one best supporter**

$$g_{\text{SL}}(\mathbf{y}) = \prod_i s_i(y_i) \left[ \max_j s_{ij}(y_i, y_j) \right]$$

- ▶  $\mathbf{y}$  is the vector of entity labels assigned to all mentions in a document
- ▶  $s_i(y_i)$  is the local score for entity label  $y_i$  for mention/spot  $i$
- ▶ MAP inference is still intractable
  - ▶ If  $j$  is the best supporter of  $i$ , is  $i$  necessarily the best supporter of  $j$ ?
- ▶ Approximate by message passing (loopy belief propagation) on factor graph
- ▶ Factor  $a_i$  for each mention  $i$

## Single link baseline (2)

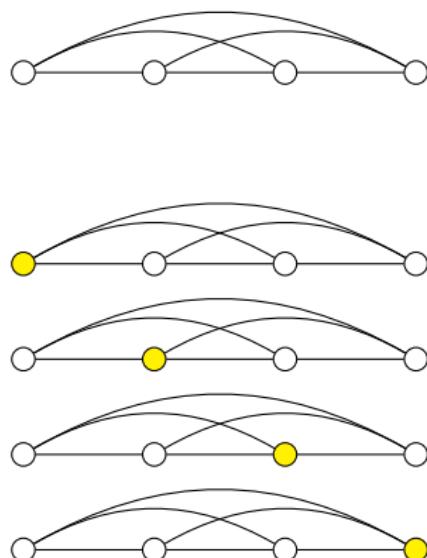
- ▶ Each factor connects to all (mention) nodes, but best supporter makes message passing practical
- ▶ Message from  $a_k$  to mention  $i$  is

$$n_{a_k \rightarrow i}(y_i) = \max_{\mathbf{y}_{\setminus i}} \left[ \psi_k(y_i, \mathbf{y}_{\setminus i}) \prod_{j \neq i} m_{j \rightarrow a_k}(y_j) \right]$$

- ▶ Belief in  $\mathbf{y}$  based on incoming messages from all factors

## Relaxing to a star model

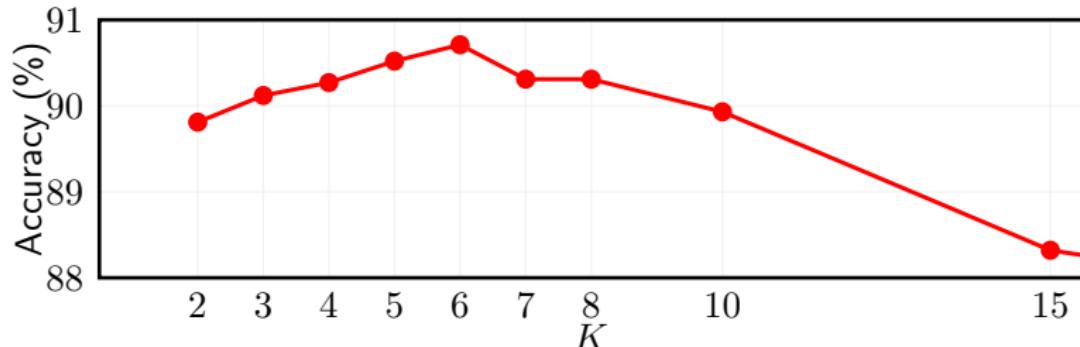
- ▶ Give up global consistency for tractability
- ▶ In turn, make each mention center of a star
- ▶ Assign label to each spoke separately to maximize support for hub
- ▶ Support for label  $y_i$  from mention  $j$  is  
$$q_{ij}(y_i) = \max_{y_j} [s_{ij}(y_i, y_j) + s_j(y_j)]$$
- ▶ Score function for mention  $i$  is  
$$f_i(y_i) = s_i(y_i) + \sum_{\text{all } j \neq i} q_{ij}(y_i)$$
- ▶ Predict  $y_i$  by maximizing above score
- ▶ Next step: replace  $\text{all } j \neq i$  with something more robust
- ▶ In what follows, let  
$$\mathbf{q}_i(y_i) = \langle q_{i1}(y_i), \dots, q_{in}(y_i) \rangle$$
 be the sequence of support from other mentions to mention  $i$



## You need only six good friends

- ▶ Star model with top- $K$  supporters:
  - ▶ When choosing  $e_i$ , set other  $e_j$  to get the best top- $K$  supporters  $e_j$ , rather than all  $n - 1$
  - ▶ Later, when setting  $e_j$ , do not constrain  $e_i$  to be the label earlier chosen
- ▶ Best support for label  $e_i$  from mention  $j$  is  
$$q_{ij}(e_i) = \max_{e_j} [\Psi(e_j) + \Phi(e_i, e_j)]$$
- ▶ Star model with all  $n - 1$  supporters amounts to overall score  
$$f_i(e_i) = \Psi(e_i) + \sum_{j \neq i} q_{ij}(e_i)$$
- ▶ Let  $\mathbf{q}_i(e_i) = \langle q_{i1}(e_i), \dots, q_{in}(e_i) \rangle$  be the sequence of supports from other mentions to mention  $i$
- ▶ Given support sequence  $\mathbf{q}$ , let  $\text{amx}_K(\mathbf{q})$  be the sum of the largest  $K$  elements of  $\mathbf{q}$
- ▶ Redefine score function for  $i$ th mention as  
$$f_i(e_i) = \Psi(e_i) + \text{amx}_K(\mathbf{q}_i(e_i))$$

## You need only six good friends (2)



- ▶ Plot accuracy against  $K$
- ▶ Single supporter too little to go by
- ▶ All  $n - 1$  supporters too much to ask for
- ▶ Clear peak at  $K = 6$
- ▶  $K$  supporters get full backprop, others get none
- ▶ From  $K$ -max to soft- $K$ -max

## Multifocal last step: from max to soft-max

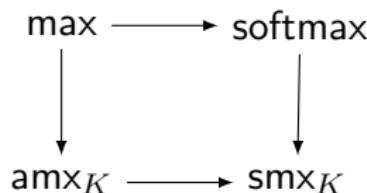
- ▶ Find maximum element in non-negative vector  $q$  is equivalent to  $\max_{u \in \Delta} u \cdot q$
- ▶  $\Delta$  is the unit simplex
- ▶  $u$  will concentrate on one corner of  $\Delta$
- ▶ Anneal with entropy:  $\max_{u \in \Delta} u \cdot q + H(u)/\beta$
- ▶ Easy to see solution as  $u_i \propto \exp(\beta q_i)$
- ▶ In other words, adding entropic annealing to max gives us soft-max
- ▶ In standard multiclass classification, benefit of soft-max is continuous differentiability
- ▶ Can backprop to downstream model components

## Soft multifocal attention

- ▶ Recall  $\mathbf{q} = \langle q_{i1}(y_i), \dots, q_{in}(y_i) \rangle$  is the vector of supports for  $y_i$
- ▶ Add entropy term to **amx** to get **smx**:

$$\text{smx}_K(\mathbf{q}) = \max_{\mathbf{u} \in \Delta_K} \left[ \mathbf{q} \cdot \mathbf{u} - \frac{1}{\beta} \sum_i u_i \log u_i \right]$$

- ▶ Here  $\Delta_K$  is the  $K$ -simplex:  $\mathbf{u} \geq \vec{0}$  and  $\|\mathbf{u}\|_1 = K$
  - ▶  $\text{smx}_K$  can be computed easily and is differentiable
- ▶ HW Apply to fine typing and other applications where softmax gives excessively skewed attention



## Soft multifocal attention

- ▶ Note  $\text{amx}_K(\mathbf{q}) = \max_{\vec{\mathbf{z}} \leq \vec{\mathbf{1}}} \mathbf{z} \cdot \mathbf{q}$  s.t.  $\sum_j z_j = K$
- ▶ Replace  $\text{amx}_K$  with **soft K-max**  
 $\text{smx}_K(\mathbf{q}) = \max_{\vec{\mathbf{z}} \leq \vec{\mathbf{1}}} \mathbf{z} \cdot \mathbf{q} - \underbrace{\sum_j z_j \log z_j}_{\text{entropy}}$  s.t.  $\sum_j z_j = K$
- ▶ Generalizes softmax
- ▶ Used to train model weights inside  $\Psi, \Phi$

System	Alias-entity map	Accuracy%
Lazic+ 2015	Older KG	86.4
Our baseline	Latest KG	87.9
Single link	Latest KG	88.2
Multifocal	Latest KG	89.5
Chisholm+ 2015	YAGO	88.7
Our baseline	YAGO+KG	85.2
Single link	YAGO+KG	86.6
Multifocal	YAGO+KG	91.0
Multifocal	KG+HP	92.7

## Soft multifocal attention (2)

- ▶ Within each alias-entity map, single-link and multifocal are the best
- ▶ Baseline and single link degrade when alias map changes from KG to YAGO+KG (larger ambiguity), but multifocal improves
- ▶ Similar consistent gains in TAC 2010, 2011, 2012
- ▶ What's missing? Entity embeddings

## Using entity embeddings [13]

- ▶ Three-part optimization
- ▶ Overall likelihood fitted through simultaneous maximization

$$\mathcal{L} = \mathcal{L}_w + \mathcal{L}_e + \mathcal{L}_a$$

Word-word:  $\mathcal{L}_w$ , standard word2vec on text corpus

Entity-entity:  $\mathcal{L}_e$ , as expressed through KG

Word-entity:  $\mathcal{L}_a$ , connecting mention context words and entity embeddings

- ▶  $e, e'$  are related if there is a link between them in the KG, and  $e \neq e'$ , in which case we want large

$$\mathcal{L}_e = \sum_{e, e'} \log \Pr(e' | e), \quad \text{where}$$

$$\Pr(e' | e) = \frac{\exp(\mathbf{u}_e \cdot \mathbf{v}_{e'})}{\sum_e \exp(\mathbf{u}_e \cdot \mathbf{v}_e)}$$

## Using entity embeddings [13] (2)

- ▶ As in skip-gram, predict mention context words given focus entity ID
- ▶ Let  $M_e$  be mentions of entity  $e$ ,  $m \in M_e$  be one mention, and  $w \in m$  a mention word

$$\mathcal{L}_a = \sum_e \sum_{m \in M_e} \sum_{w \in m} \log \Pr(w|e),$$

where  $\Pr(w|e) = \frac{\exp(\mathbf{v}_w \cdot \mathbf{u}_e)}{\sum_{w'} \exp(\mathbf{v}_{w'} \cdot \mathbf{u}_e)}$

- ▶ As is common, softmax is replaced by negative samples

## Inference with coherence

- ▶ Given a document with many mention spots
- ▶ For each mention, compute context vector as average of neighboring word vectors
- ▶ (Nothing more fancy like convnet or RNN)
- ▶ Set initial entity labels using cosine with context vectors
- ▶ Now define the coherence of an entity with others as average cosine between entity vectors
- ▶ Reassign most coherent label in a second step
- ▶ Crude two-step loopy BP?

## Joint word-entity embeddings: NED results

	CoNLL (Micro)	CoNLL (Macro)	TAC10 (Micro)
Yamada et al., 2016	<b>93.1</b>	<b>92.6</b>	<b>85.2</b>
Hoffart et al., 2011	82.5	81.7	-
He et al., 2013	85.6	84.0	81.0
Chisholm & Hachey, 2015	88.7	-	80.7
Pershina et al., 2015	91.8	89.9	-

## Attention on mention context [14]

- ▶ Jointly pre-embed all words  $w$  and entities  $e$  in training corpus (Wikipedia, say) to (focus) embeddings  $x_w, x_e$
- ▶ Given a mention  $m$  with candidates  $\Gamma(m)$ , mention context  $c$  mentioning entity  $e \in \Gamma(m)$ , for each word  $w$  in the context, compute the importance of  $w$  as

$$u(w) = \max_{e \in \Gamma(m)} x_e^\top \mathbf{A} x_w,$$

where  $\mathbf{A}$  is a global (diagonal) matrix to be trained

- ▶ Intention:  $u(w)$  should be large if  $w$  is strongly associated with at least one candidate entity, otherwise small
- ▶ Sort by decreasing  $u(w)$  and prune context to top- $K$
- ▶ Now let surviving context words compete for attention:

$$\beta(w) = \exp(u(w)) / \sum_{w'} \exp(u(w'))$$

## Attention on mention context [14] (2)

- ▶ Compute similarity between  $x_e$  and  $x_w$  and add up, weighted by attention:

$$\Psi(e, c) = \sum_w \beta(w) x_e^\top \mathbf{B} x_w ,$$

where  $\mathbf{B}$  is another global diagonal matrix to be trained

- ▶ Note, very frugal model so far, only  $2D$  model weights, where embeddings are in  $\mathbb{R}^D$
- ▶ Finally, combine with (empirical) mention prior  $\Pr(e|m)$ :

$$\Psi(e, m, c) = N(\Psi(e, c), \log \Pr(e|m)),$$

where  $N$  is a 2-layer fully-connected network with 100 hidden units and ReLU nonlinearities

## Attention on mention context [14] (3)

- ▶ For training, use standard hinge loss

$$\operatorname{argmin}_{\mathbf{A}, \mathbf{B}, N, \dots} \sum_m \sum_{e \in \Gamma(m)} [\clubsuit - \Psi(e^*, m, c) + \Psi(e, m, c)]_+,$$

where  $\clubsuit$  is a tuned margin

- ▶ Local attention model results:

Methods	AIDA-test-b
Mention prior $\Pr(e m)$	71.9
(Lazic et al., 2015)	86.4
(Yamada et al., 2016)	87.2
(Globerson et al., 2016)	87.9
Ganea+ (local, K=100, R=50)	<b>88.8</b>

- ▶ Network  $N$  benefits from nonlinearity

## Document-level deep model

- ▶ Now we bring back in global coherence between entity labels
- ▶ For a whole document, let  $\mathbf{e}, \mathbf{m}, \mathbf{c}$  be the sequence of  $n$  entity labels, mentions, and contexts
- ▶ Fully connected pairwise random field

$$g(\mathbf{e}, \mathbf{m}, \mathbf{c}) = \frac{1}{n} \sum_i \Psi(e_i, m_i, c_i) + \frac{1}{\binom{n}{2}} \sum_{i < j} \Phi(e_i, e_j),$$

where  $\Phi(e, e') = \mathbf{x}_e^\top \mathbf{C} \mathbf{x}_{e'}$

- ▶ Note, all mention pairs
- ▶  $\mathbf{C}$  is another diagonal weight matrix to be trained
- ▶ Inference amounts to finding  $\operatorname{argmax}_{\mathbf{e}} g(\mathbf{e}, \mathbf{m}, \mathbf{c})$ , given observed  $\mathbf{m}, \mathbf{c}$
- ▶ Back to (max-product) message-passing

## Document-level deep model (2)

- ▶ In iteration  $t$ , mention  $m_i$  votes for entity candidate  $e' \in \Gamma(m_j)$  using outgoing (log) message

$$m_{i \rightarrow j}^{t+1}(e') = \max_{e \in \Gamma(m_i)} \left[ \Psi(e, m_i, c_i) + \Phi(e, e') + \sum_{k \neq j} \overline{m}_{k \rightarrow i}^t(e) \right]$$

- ▶ The incoming messages would ordinarily be just log-beliefs:

$$\overline{m}_{i \rightarrow j}^t(e) = \log \text{softmax}(m_{i \rightarrow j}^t(e))$$

- ▶ In practice, **damping** with  $\delta \in (0, 1]$  helps stability and convergence:

$$\overline{m}_{i \rightarrow j}^t(e) = \log \left[ \delta \text{softmax}(m_{i \rightarrow j}^t(e)) + (1 - \delta) \exp(\overline{m}_{i \rightarrow j}^{t-1}(e)) \right]$$

## Document-level deep model (3)

- ▶ Unroll BP to  $T$  time steps, resulting in final beliefs

$$\mu_i(e) = \Psi(e, m_i, c_i) + \sum_{k \neq i} \overline{m}_{k \rightarrow i}^T(e)$$
$$\overline{\mu}_i(e) = \frac{\exp(\mu_i(e))}{\sum_{e' \in \Gamma(m_i)} \exp(\mu_i(e))}$$

- ▶ Given the above inference procedure, we can use it for training as well
- ▶ Given gold entity labels, express hinge loss wrt final beliefs:

$$\operatorname{argmin}_{\mathbf{A}, \mathbf{B}, \mathbf{C}, N} \sum_m \sum_{e \in \Gamma(m)} [\spadesuit - \overline{\mu}_i(e^*) + \overline{\mu}_i(e)]_+$$

- ▶ Hinge loss assessed wrt per-variable marginals
- ▶ Everything is still end-to-end (sub)differentiable 😊

## Document-level deep model (4)

- ▶ May be simpler (but possibly less accurate) than sampling negative instances and expressing objective as hinge loss corresponding to

$$\forall \mathbf{e}_- : \quad g(\mathbf{e}_+, \mathbf{m}, \mathbf{c}) \geq g(\mathbf{e}_-, \mathbf{m}, \mathbf{c}) + \spadesuit$$

## Ganea et al.: global results

Global method	AIDA-test-b
(Huang et al., 2015)	86.6
(Ganea et al., 2016)	87.6
(Chisholm and Hachey, 2015)	88.7
(Guo and Barbosa, 2016)	89.0
(Globerson et al., 2016)	91.0
(Yamada et al., 2016)	91.5
Ganea+ (global)	<b>92.22±0.14</b>

- ▶ Impressive gains with very few model weights!
- ▶ Even more impressive that tail entities work out so well
- ▶ OTOH the whole network is quite complex; quite a wonder that backprop through such hostile functions works so well to depth  $O(T)$
- ▶ Many potential bad choices for  $A, B, N$ ; would be good to know how robust the design is

## References

- [1] A. Sil and A. Yates, "Re-ranking for joint named-entity recognition and linking," in *CIKM*, 2013, pp. 2369–2374. [Online]. Available: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.398.9086&rep=rep1&type=pdf>
- [2] R. Bunescu and M. Pasca, "Using encyclopedic knowledge for named entity disambiguation," in *EACL*, 2006, pp. 9–16. [Online]. Available: <http://www.cs.utexas.edu/~ml/papers/encyc-eacl-06.pdf>
- [3] J. Hoffart *et al.*, "Robust disambiguation of named entities in text," in *EMNLP Conference*. Edinburgh, Scotland, UK: SIGDAT, Jul. 2011, pp. 782–792. [Online]. Available: <http://aclweb.org/anthology/D/D11/D11-1072.pdf>
- [4] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti, "Collective annotation of Wikipedia entities in Web text," in *SIGKDD Conference*, 2009, pp. 457–466. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/CSAW/>

## References (2)

- [5] S. Dill *et al.*, "SemTag and Seeker: Bootstrapping the semantic Web via automated semantic annotation," in *WWW Conference*, 2003, pp. 178–186.
- [6] R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *CIKM*, 2007, pp. 233–242. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1321440.1321475>
- [7] D. Milne and I. H. Witten, "Learning to link with Wikipedia," in *CIKM*, 2008, pp. 509–518. [Online]. Available: <http://www.cs.waikato.ac.nz/~dnk2/publications/CIKM08-LearningToLinkWithWikipedia.pdf>
- [8] X. Cheng and D. Roth, "Relational inference for wikification," in *EMNLP Conference*, 2013, pp. 16–58. [Online]. Available: <https://www.aclweb.org/anthology/D/D13/D13-1184.pdf>
- [9] M. Ponza, P. Ferragina, and S. Chakrabarti, "A two-stage framework for computing entity relatedness in wikipedia," in *CIKM*, 2017, pp. 1867–1876. [Online]. Available: <https://dl.acm.org/citation.cfm?id=3132890>

## References (3)

- [10] S. Cucerzan, "Large-scale named entity disambiguation based on Wikipedia data," in *EMNLP Conference*, 2007, pp. 708–716. [Online]. Available: <http://www.aclweb.org/anthology/D/D07/D07-1074>
- [11] L. Ratinov, D. Roth, D. Downey, and M. Anderson, "Local and global algorithms for disambiguation to Wikipedia," in *ACL Conference*, ser. ACL/HLT, Portland, Oregon, 2011, pp. 1375–1384. [Online]. Available: <http://dl.acm.org/citation.cfm?id=2002472.2002642>
- [12] A. Globerson, N. Lazic, S. Chakrabarti, A. Subramanya, M. Ringgaard, and F. Pereira, "Collective entity resolution with multi-focal attention," in *ACL Conference*, 2016, pp. 621–631. [Online]. Available: <https://www.aclweb.org/anthology/P/P16/P16-1059.pdf>
- [13] I. Yamada, H. Shindo, H. Takeda, and Y. Takefuji, "Joint learning of the embedding of words and entities for named entity disambiguation," *arXiv preprint arXiv:1601.01343*, 2016. [Online]. Available: <https://arxiv.org/pdf/1601.01343.pdf>

## References (4)

- [14] O.-E. Ganea and T. Hofmann, "Deep joint entity disambiguation with local neural attention," *arXiv preprint arXiv:1704.04920*, 2017. [Online]. Available: <https://arxiv.org/pdf/1704.04920.pdf>

## Coreference resolution and record linkage

## Two major domains

- ▶ Coref resolution in natural language text
  - ▶ Linear text source with spots/mentions
  - ▶ References to entities, may include pronouns
  - ▶ Usually turn spots into feature vectors
  - ▶ Cluster feaure vectors into mentions of same entity
- ▶ Record linkage in data warehousing
  - ▶ Record linkage, deduplication, merge-purge
  - ▶ Record may be obtained via tagging/extraction
  - ▶ Clear schema, attributes, structured values
  - ▶ Goal: cluster records into equivalence classes

## Example: paper citations

- ▶ (MLA) Lafferty, John, Andrew McCallum, and Fernando Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data." Proceedings of the eighteenth international conference on machine learning, ICML. Vol. 1. 2001.
- ▶ (APA) Lafferty, J., McCallum, A., & Pereira, F. (2001, June). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the eighteenth international conference on machine learning, ICML (Vol. 1, pp. 282-289).
- ▶ (Chicago) Lafferty, John, Andrew McCallum, and Fernando Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data." In Proceedings of the eighteenth international conference on machine learning, ICML, vol. 1, pp. 282-289. 2001.
- ▶ (Harvard) Lafferty, J., McCallum, A. and Pereira, F., 2001, June. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Proceedings of the eighteenth international conference on machine learning, ICML (Vol. 1, pp. 282-289).
- ▶ (Vancouver) Lafferty J, McCallum A, Pereira F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. InProceedings of the eighteenth international conference on machine learning, ICML 2001 Jun 28 (Vol. 1, pp. 282-289).

## Types of coref<sup>6</sup>

- Anaphora ► The music was so loud that it couldn't be enjoyed.
  - Our neighbors dislike the music. If they are angry, the cops will show up soon.
- Cataphora ► If they are angry about the music, the neighbors will call the cops.
  - Despite her difficulty, Wilma came to understand the point.
- Split antecedents ► Carol told Bob to attend the party. They arrived together.
  - When Carol helps Bob and Bob helps Carol, they can accomplish any task.
- Coreferring noun phrases ► The project leader is refusing to help.
  - The jerk thinks only of himself.
  - Some of our colleagues are going to be supportive. These kinds of people will earn our gratitude.

---

<sup>6</sup>From Wikipedia

## Coref difficulties

- ▶ Pronouns are an important coref device, but one among many
- ▶ The man who put his money in his bank was wiser than the man who spent it on his cars.
- ▶ Other anaphoric relations: The boy entered the room. The door closed automatically.
- ▶ A more complex sequence of sentences:
  - ▶ Terry really goofs sometimes.
  - ▶ Yesterday was a beautiful day and he was excited about trying out his new sailboat.
  - ▶ He wanted Tony to join him on a sailing expedition.
  - ▶ He called him at 6 AM.
  - ▶ He was sick and furious at being woken up so early.
- ▶ You only die once: After he died, Obama praised Ariel Sharon.
- ▶ The ball smashed the table to pieces because it was made of {balsa  
steel}.
- ▶ Pleonastic "it": It's raining. It takes a lot of work.

## Coref as classification

- ▶ In general, we need to link a **referent** to an **antecedent**, most often before but sometimes after the referent
- ▶ Assume referent token spans have been identified in a doc
- ▶ E.g., might be all pronouns
- ▶ For each referent, assume all candidate antecedent spans have been identified
- ▶ E.g., *he* and *she* to all gender-compatible person mentions
- ▶ Use a number of features to predict the antecedent:
  - ▶ Distance: number of tokens, edges on dependency path, etc.
  - ▶ Syntactic structure
  - ▶ Agreement in gender, number and **animacy**<sup>7</sup>
  - ▶ Mention count of antecedent

Excellent IJCAI 2016 tutorial on coref

---

<sup>7</sup>Animacy: How sentient or alive the referent of a noun is

## Record linkage

- ▶ No catalog, or we know spots do not correspond to any element in catalog
- ▶ Cluster spots/mentions so that each cluster refers to distinct real-world entity
- ▶ Common application: collect material about a specific person on the Web, avoiding namesakes
- ▶ Structured vs. unstructured sources
  - ▶ Telephone directory, income tax database, municipal property database
  - ▶ Vs. mentions embedded in unstructured text
- ▶ Pairwise decisions and transitive consistency
  - ▶ Each edge marked “+” or “−” and has weight  $w_{ij}$
  - ▶ Pay  $w_{ij}$  if global partitioning differs from pairwise judgment  $\pm(i, j)$

## Record linkage (2)

- ▶ Proposed optimization:

$$\begin{aligned} \min_{\{x_{ij}\}} \quad & \sum_{+(i,j)} w_{ij} x_{ij} + \sum_{-(i,j)} w_{ij} (1 - x_{ij}) \quad \text{s.t.} \\ & x_{ik} \leq x_{ij} + x_{jk} \quad \forall i, j, k \\ & x_{ij} \in \{0, 1\} \quad \forall i, j \end{aligned}$$

( $x_{ij} = 0$  means  $i, j$  in *same* partition)

- ▶ Iteratively learning similarity of items and relative importance of features

## Application: resolving relations

- ▶ Earlier we saw how raw tuples (triples) of the form  $(r, e_1, e_2)$  were extracted from unstructured text
- ▶  $r, e_1, e_2$  are mentions of a binary relation, and two related entities
- ▶ Examples:
  - ▶ (lacks, Mars, ozone layer)
  - ▶ (lacks, Red Planet, ozone layer)
  - ▶ (is capital of, D.C., United States)
  - ▶ (is capital city of, Washington, U.S.)

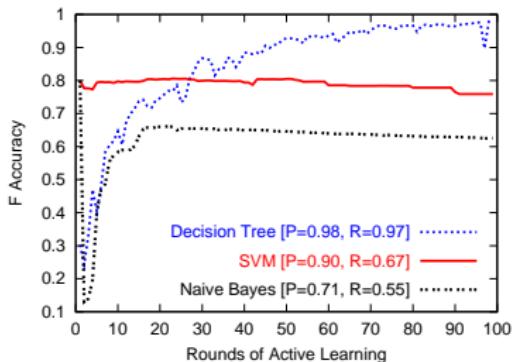
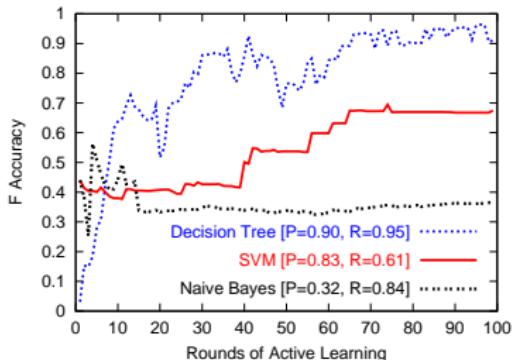
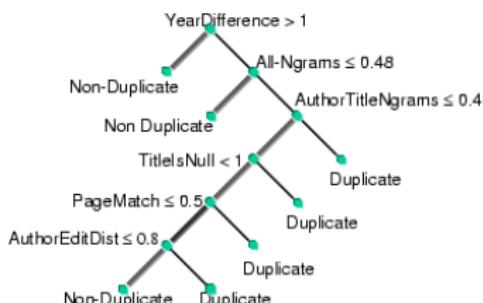
## Independent classification

- ▶ Given records or mentions  $x_1, x_2$
- ▶ Design feature vector  $\phi(x_1, x_2)$
- ▶ Learn a classifier  $f$
- ▶ E.g.,  $f(x_1, x_2) = \text{sign}(w \cdot \phi(x_1, x_2))$
- ▶ (More complicated decision surfaces usually needed)
- ▶  $+1$  means same cluster,  $-1$  means different
- ▶ How about transitivity?
- ▶ I.e., what about  $f(x_1, x_2) = f(x_2, x_3) = 1$  but  $f(x_1, x_3) = -1$ ?
- ▶ Consistent by design or patch afterwards

## Design of $f(x_1, x_2)$

- ▶ Using domain knowledge, design (dis)similarity features between corresponding attributes
- ▶ Examples for bibliographic citations
- ▶ Difference in years of publication
- ▶ Difference in (begin, end) page numbers
- ▶ TFIDF cosine similarity between paper titles
- ▶ Jaccard similarity between author token sets
- ▶ Sample duplicate and non-duplicate pairs  $x_1, x_2$
- ▶ Note most pairs non-duplicate; stratified sampling may be needed
- ▶ Compute feature vector and submit to decision tree

# Decision tree example

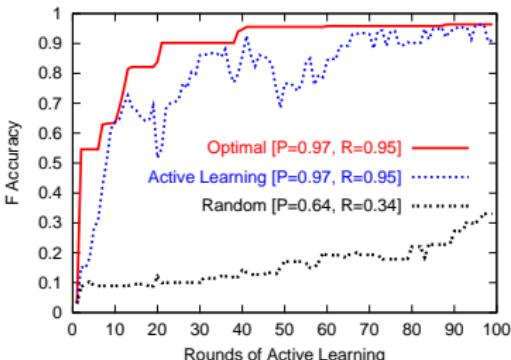


## Active learning of $f(x_1, x_2)$

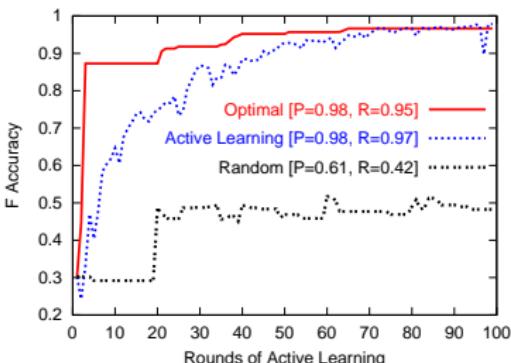
- ▶ When the decision tree classifies a new pair, what is its confidence?
- ▶ Use a **committee** of decision trees
  - ▶ Sample features
  - ▶ Sample pairs
  - ▶ Randomly perturb decision boundaries
- ▶ Disagreement  $\implies$  low confidence
- ▶ Repeatedly choose low-confidence pairs
- ▶ Ask for labels from human trainer

# Active learning results

- ▶ Random: random unlabeled pair picked
- ▶ Optimal: omniscient best next pair picked
- ▶ Active learning via decision tree committee effective at boosting test accuracy with very small number of labels
- ▶ Note, the pair vs. transitive consistency issue is still to be solved



(a) Bibliography data



(b) Address data

## Correlation clustering

- ▶ Graph with items as nodes, similarities and dissimilarities as edges
- ▶ For simplicity assume a complete graph
- ▶ Each edge marked '+' (similar) or '-' (dissimilar)
- ▶ Objective is to partition nodes so that ...
- ▶ Each + edge is within a partition
- ▶ Each - edge goes across two partitions
- ▶ If not feasible, count the number of satisfied or unsatisfied edges as the objective
- ▶ Note that the **number of clusters** is not specified

## Some simple observations

- ▶ If a solution satisfying all edges exists, it is trivial to find
- ▶ Simply delete all – edges, and find connected components using the + edges
- ▶ Otherwise . . .
- ▶ If there are more + edges than – edges, place all nodes in a single partition
- ▶ Otherwise place each node in a separate partition
- ▶ Satisfies at least half the edges
- ▶ Therefore, the number of **unsatisfied** edges may be a better objective
- ▶ While exactly optimizing satisfied and unsatisfied edges are equivalent, approximations are not

## Approximate algorithm: Cautious

- ▶ Let  $N^+(v), N^-(v)$  be neighbors of  $v$  connected through  $+, -$  edges
- ▶ For  $C \subseteq V$ , node  $v$  is  **$\delta$ -good** (not  **$\delta$ -bad**) wrt  $C$  if
  - ▶  $|N^+(v) \cap C| \geq (1 - \delta)|C|$
  - ▶  $|N^+(v) \cap (V \setminus C)| \leq \delta|C|$
- ▶ The following algorithm keeps at most  $(9/\delta^2 + 5)$  times as many edges unsatisfied as the optimal solution (proof quite involved)

**repeat**

    pick arbitrary node  $v$  and let  $A(v) = N^+(v)$

**while**  $\exists x \in A(v)$  such that  $x$  is  $3\delta$ -bad wrt  $A(v)$  **do**

        remove  $x$  from  $A(v)$

**for** each  $y$  that is  $7\delta$ -good wrt  $A(v)$  **do**

        add  $y$  to  $A(v)$

    report  $A(v)$  as a partition and remove from  $V$

**until**  $A(v)$  ends up empty

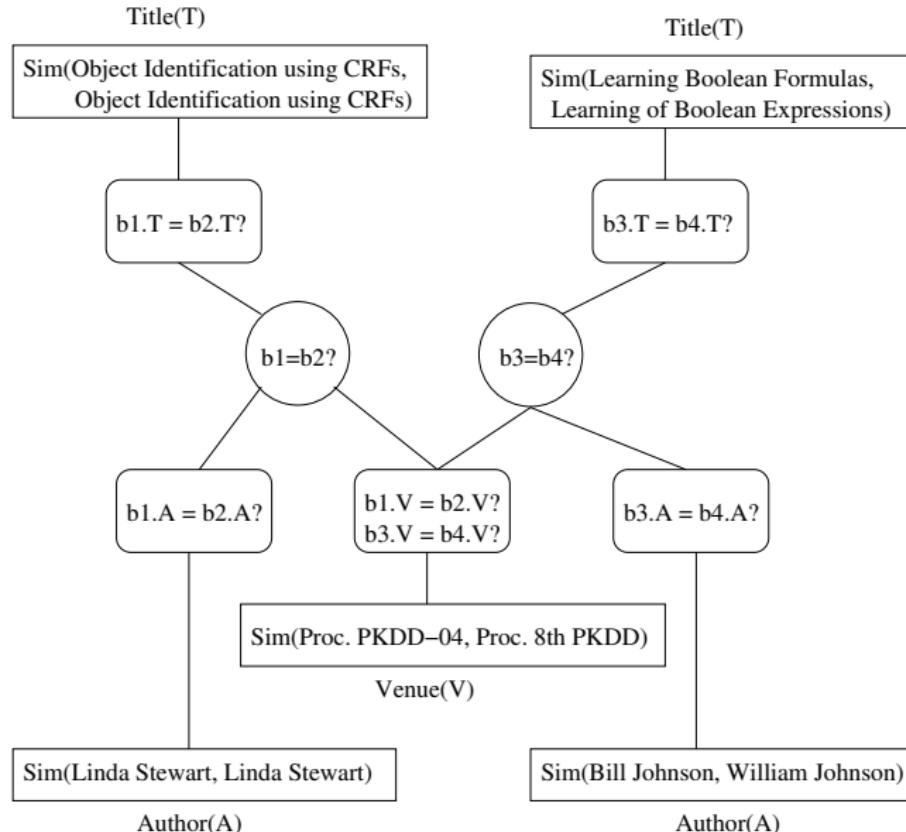
report remaining nodes as singleton partitions

## Attribute mediated resolution

Record	Title	Author	Venue
b1	Object Identification using CRFs	Linda Stewart	Proc. PKDD-04
b2	Object Identification using CRFs	Linda Stewart	Proc. 8th-PKDD
b3	Learning Boolean Formulas	Bill Johnson	Proc. PKDD-04
b4	Learning of Boolean Expressions	William Johnson	Proc. 8th-PKDD

- ▶ Suppose we figure out that b1 and b2 are the same paper
- ▶ This implies that Proc. PKDD-04 and Proc. 8th PKDD are the same conference
- ▶ Which in turn allows us to merge b3 and b4
- ▶ Which then lets us conflate Bill Johnson and William Johnson
- ▶ How to represent this formally?

# Graphical model for attribute mediation



## Model summary

- ▶ **Field-match nodes**  $F_{ab}^k$ : decision node for duplicate status of values  $a$  and  $b$  for the  $k$ th attribute
- ▶ Connected to **field-similarity nodes**  $S_{ab}^k$  holding similarity score/s between values  $x$  and  $y$  for the  $k$ th attribute
- ▶ Similarity assumed to be in  $[0, 1]$
- ▶ Also connected to **record-match nodes**  $R_{ij}$ : duplicate decision node between records  $i$  and  $j$
- ▶ Node  $R_{ij}$  is connected to each node  $F_{ab}^k$  if  $[i, k] = a$  and  $[j, k] = b$
- ▶ Recall conditional random fields notation:  $c$  is a clique,  $f$  is a feature index,  $\phi_{cf}$  is a (scalar) feature,  $\lambda_{cf}$  corresponding model parameter,  $Z_{\vec{x}}$  a partition function

$$\Pr(\vec{y}|\vec{x}) = \frac{1}{Z_{\vec{x}}} \prod_c \exp\left(\sum_f \lambda_{cf} \phi_{cf}(\vec{x}_c, \vec{y}_c)\right)$$

## Model summary (2)

- ▶ Common to tie parameters across a **clique template**
- ▶ I.e., use  $\phi_{tf}$  and  $\lambda_{tf}$
- ▶  $C_t$  is the set of cliques that use parameters from template  $t$
- ▶ Conditional probability expression modified to

$$\Pr(\vec{y}|\vec{x}) = \frac{1}{Z_{\vec{x}}} \prod_t \prod_{c \in C_t} \exp\left(\sum_f \lambda_{tf} \phi_{tf}(\vec{x}_c, \vec{y}_c)\right)$$

- ▶ In our case observed  $\vec{x}$  consists of the similarity nodes  $\vec{s}$  and we need to infer  $\vec{r}$  and  $\vec{f}$ : record and field match decisions
- ▶ It remains to instantiate  $\lambda$  and  $\phi$  for our application
- ▶ Nodes are labeled with uppercase  $R, F$  etc., specific values are written  $r, f$
- ▶ Let  $[i, k]$  denote column/attribute  $k$  of row  $i$  in the (unnamed) table we are deduping

## Model summary (3)

- We will multiply three kinds of terms:

$$\text{(Record-prior)} \quad \prod_{i,j} \exp(\lambda^1 \cdot \phi^1(r_{ij}))$$

$$\text{(Record-field)} \quad \prod_{i,j,k} \exp(\lambda^2 \cdot \phi^2(r_{ij}, f_{[i,k][j,k]}^k))$$

$$\text{(Field-score)} \quad \prod_{k,a,b} \exp(\lambda^3 \cdot \phi^3(f_{ab}^k, s_{ab}^k))$$

- Technicality: Singla and Domingos consider each  $a, b$  as many times as mentioned in tuples (sort of like microaveraging) instead of just once like above (like macroaveraging)
- Math remains entirely comparable
- Superscript in  $\lambda$  and  $\phi$  separates different blocks of parameters and features

## Feature design

- ▶ To capture **marginal probability** of duplicate vs. non-duplicate, use (redundant) parameters  $\lambda_0^1, \lambda_1^1$  and corresponding features

$$\phi_0^1(r_{ij}) = \begin{cases} 1, & r_{ij} = 0 \\ 0, & r_{ij} = 1 \end{cases} \quad \phi_1^1(r_{ij}) = \begin{cases} 0, & r_{ij} = 0 \\ 1, & r_{ij} = 1 \end{cases}$$

- ▶ Coupling between  $R_{ij}$  and  $F_{ab}^k$  is expressed by features  $\phi^2(r_{ij}, f_{ab}^k)$  with corresponding model parameters  $\lambda^2$
- ▶ Four possible inputs to  $\phi^2$ : 00, 01, 10, 11
- ▶ Can select each of these as  $\phi_{00}^2(r, f) = \llbracket r = 0 \wedge f = 0 \rrbracket$  etc.
- ▶ With four parameters  $\lambda_{00}^2, \lambda_{01}^2, \lambda_{10}^2, \lambda_{11}^2$
- ▶ (These are the key parameters)
- ▶ Can diversify by column  $k$  as well, i.e.,  $\lambda_{k00}^2, \lambda_{k01}^2, \lambda_{k10}^2, \lambda_{k11}^2$

## Feature design (2)

- ▶ Coupling between  $S_{ab}^k$  and  $F_{ab}^k$  is expressed using node features  $\phi^3(s_{ab}^k, f_{ab}^k)$  and parameters  $\lambda^3$
- ▶ Singla et al. use

$$\phi_0^3(s, f) = \begin{cases} 1 - s, & f = 0 \\ 0, & f = 1 \end{cases} \quad \phi_1^3(s, f) = \begin{cases} 0, & f = 0 \\ s, & f = 1 \end{cases}$$

- ▶ With corresponding parameters  $\lambda_0^3, \lambda_1^3$
- ▶ Again, can diversify with  $k$ , as in  $\lambda_{k0}^3, \lambda_{k1}^3$

## Binary MRF/CRF

- ▶ A binary MRF (node labels 0/1) with 1- and 2-cliques
- ▶ Log node potential  $\varphi_i(y_i) : \{0, 1\} \rightarrow \mathbb{R}$
- ▶ Log edge potential  $\psi_{ij}(y_i, y_j) : \{0, 1\}^2 \rightarrow \mathbb{R}$
- ▶ Note, log potentials can be negative
- ▶ Objective is

$$\arg \max_{\{y_i\}} \sum_i \varphi_i(y_i) + \sum_{i,j} \psi_{ij}(y_i, y_j)$$

- ▶ In general this is NP-hard
- ▶ Intensively studied in computer vision
- ▶ There they **minimize** inversely related objectives (“energy”)
- ▶ Switching signs on  $\varphi$  and  $\psi$ , equivalent *for exact solutions*
- ▶ We will do this henceforth
- ▶ Approximations may *not* be preserved

## Inference via mincut: node potentials

- ▶ Switch sign on  $\varphi$  and  $\psi$  so that the objective is

$$\arg \min_{\vec{y}} E(\vec{y}) = \arg \min_{\{y_i\}} \sum_i \varphi_i(y_i) + \sum_{i,j} \psi_{ij}(y_i, y_j)$$

- ▶ For *binary* states, under certain conditions on  $\psi$ , this can be turned into a **mincut** problem with nonnegative edge capacities
- ▶ Create a graph with nodes 0 and 1
- ▶ Remember  $\varphi_i(0) < \varphi_i(1)$  prefers  $y_i = 0$  in min energy setting
- ▶ Therefore, for each node  $i$  ...
- ▶ If  $\varphi_i(0) \leq \varphi_i(1)$  then bias  $y_i$  toward 0 by connecting node  $i$  to 1 with directed edge weight  $\varphi_i(1) - \varphi_i(0)$
- ▶ else ( $\varphi_i(0) > \varphi_i(1)$ ) bias  $y_i$  toward 1 by connecting 0 to node  $i$  with directed edge weight  $\varphi_i(0) - \varphi_i(1)$
- ▶ Fix node  $i$  with  $\varphi_i(1) - \varphi_i(0) \geq 0$

## Inference via mincut: node potentials (2)

- ▶ If  $y_i = 0$  as per mincut, then
  - ▶ Original contribution to the objective would be  $\varphi_i(0)$
  - ▶ Modified contribution is 0
- ▶ If  $y_i = 1$  as per mincut, then
  - ▶ Original contribution to the objective would be  $\varphi_i(1)$
  - ▶ Modified contribution is  $\varphi_i(1) - \varphi_i(0)$
- ▶ **modified – original =  $\varphi_i(0)$**
- ▶ Similarly, for all  $i$  such that  $\varphi_i(1) - \varphi_i(0) < 0$ ,  
**modified – original =  $\varphi_i(1)$**
- ▶ Therefore, original and modified differ in a constant
- ▶ (Note, this is ok only because we have an *exact* algorithm for the modified problem)

## Mincut: edge potentials

- ▶ Recall  $\psi_{ij}(y_i, y_j) : \{0, 1\}^2 \rightarrow \mathbb{R}$
- ▶ Assume symmetric matrix  $\begin{bmatrix} A & B \\ B & A \end{bmatrix}$  represents negative log edge potential
- ▶ (Can generalize to wider class of matrices)
- ▶ **Associative potential** required:  $A < B$
- ▶ Note that in maximization version we would say  $A > B$
- ▶ Mapping back to dedup, if we are told some  $F_{ab}^k = 1$ , it can only increase the chances that  $R_{ij} = 1$
- ▶ Adding a constant changes nothing, so we can further simplify the potential matrix to  $\begin{bmatrix} 0 & B - A \\ B - A & 0 \end{bmatrix} = \begin{bmatrix} 0 & D \\ D & 0 \end{bmatrix}$  with  $D > 0$

## Mincut: edge potentials (2)

- ▶ This says there is no penalty for  $y_i = y_j$  but some positive penalty for  $y_i \neq y_j$
- ▶ Simply connect nodes  $i, j$  with an edge of capacity  $D$

# Training

- ▶ Example of a log-linear model

$$\Pr(\vec{y}|\vec{x}; \lambda) = \frac{\exp(\lambda \cdot \phi(\vec{x}, \vec{y}))}{Z_{\vec{x}}}$$

- ▶ Global optimum, can use hill climbing
- ▶ Can compute gradient wrt  $\lambda$  along with objective
- ▶ Must also enforce some box constraints like  $D > 0$
- ▶ Can do this through transformations like  $D = e^d$

## Attribute mediation results

Citation Matching						
Model	Before transitive closure			After transitive closure		
	F-measure	Recall	Precision	F-measure	Recall	Precision
Standard	86.9	89.7	85.3	84.7	98.3	75.5
Collective	87.4	91.2	85.1	88.9	96.3	83.3
Combined	85.8	86.1	87.1	89.0	94.9	84.5

Author Matching						
Model	Before transitive closure			After transitive closure		
	F-measure	Recall	Precision	F-measure	Recall	Precision
Standard	79.2	65.8	100	89.5	81.1	100
Collective	90.4	99.8	83.1	90.1	100	82.6
Combined	88.7	99.7	80.1	88.6	99.7	80.2

Venue Matching						
Model	Before transitive closure			After transitive closure		
	F-measure	Recall	Precision	F-measure	Recall	Precision
Standard	48.6	36.0	75.4	59.0	70.3	51.6
Collective	67.0	62.2	77.4	74.8	90.0	66.7
Combined	86.5	85.7	88.7	82.0	96.5	72.0

## References

- [1] N. Ge, J. Hale, and E. Charniak, "A statistical approach to anaphora resolution," in *Proceedings of the sixth workshop on very large corpora*, vol. 71, 1998, p. 76. [Online]. Available: <http://www.aclweb.org/anthology/W98-1119>
- [2] M. Charikar, V. Guruswami, and A. Wirth, "Clustering with qualitative information," in *FOCS Conference*, 2003, pp. 524–533. [Online]. Available: <http://www.cs.mu.oz.au/~awirth/pubs/awirthFocs03.pdf>
- [3] S. Sarawagi and A. Bhagat, "Interactive deduplication using active learning," in *SIGKDD Conference*, ser. KDD '02. New York, NY, USA: ACM, 2002, pp. 269–278. [Online]. Available: <http://www.cse.iitb.ac.in/~sunita/papers/kdd02.pdf>
- [4] N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," in *FOCS Conference*, 2002, p. 238. [Online]. Available: <http://www.cs.cmu.edu/~shuchi/papers/clusteringfull.pdf>

## References (2)

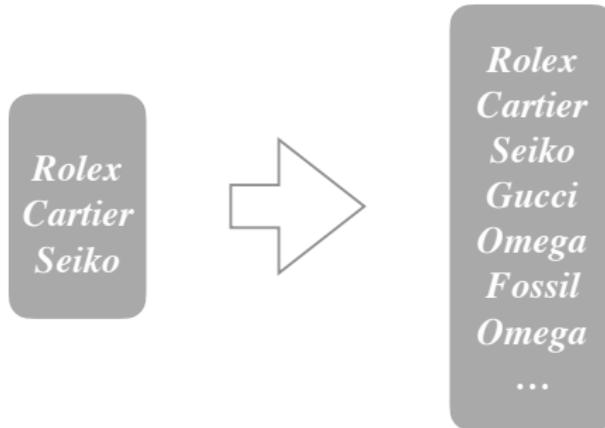
- [5] A. McCallum and B. Wellner, "Conditional models of identity uncertainty with application to noun coreference," in *NIPS Conference*, 2004, pp. 905–912. [Online]. Available: <https://papers.nips.cc/paper/2557-conditional-models-of-identity-uncertainty-with-application-to-noun-coreference.pdf>
- [6] P. Singla and P. Domingos, "Object identification with attribute-mediated dependences," in *PKDD Conference*, Porto, Portugal, 2005, pp. 297–308. [Online]. Available: <http://www.cs.washington.edu/homes/parag/papers/object-mediated-pkdd05.pdf>
- [7] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *IEEE PAMI*, vol. 26, no. 2, pp. 147–159, Feb. 2004. [Online]. Available: <http://www.cs.cornell.edu/rdz/Papers/KZ-ECCV02-graphcuts.pdf>
- [8] D. M. Greig, B. T. Porteous, and A. Seheult, "Exact maximum a posteriori estimation for binary images," *Journal of the Royal Statistical Society*, vol. B, no. 51, pp. 271–279, 1989. [Online]. Available: <http://jstor.org/stable/2345609>

# Open entity set expansion

(Includes slides from Partha Talukdar)

# Set Expansion

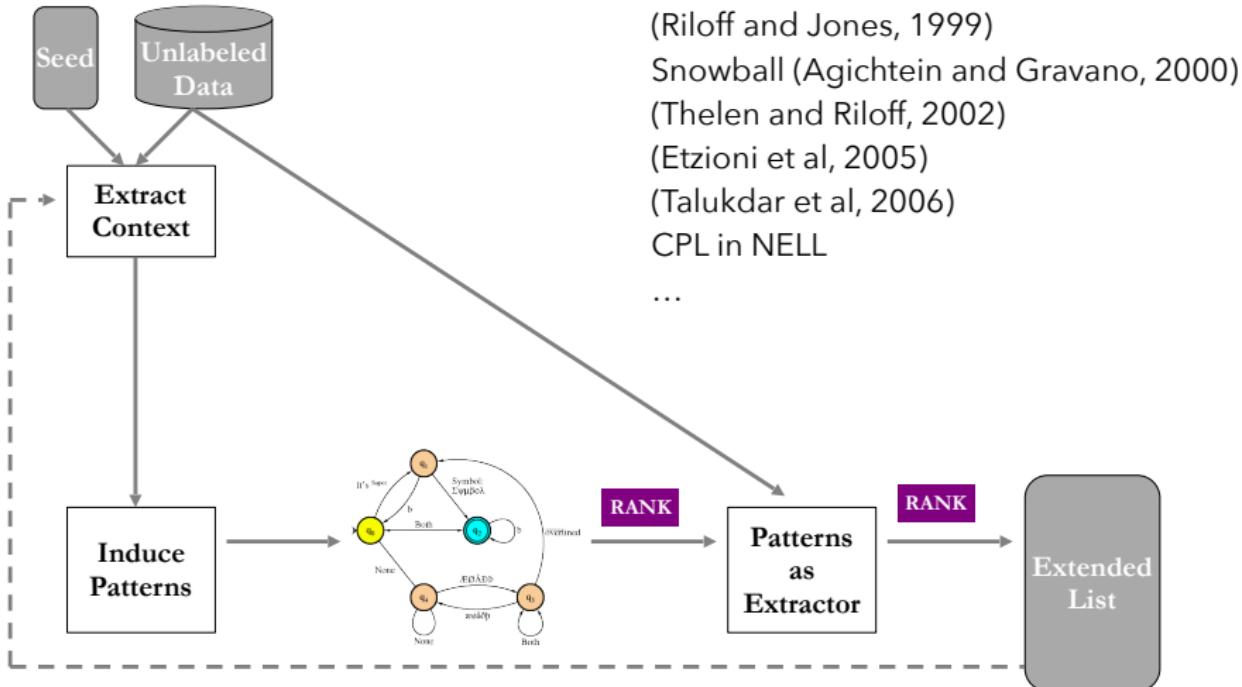
Given seed instances from a class, automatically identify more instances from that class



Many applications:

web advertising, knowledge graph population, ...

# Context Pattern Induction



## Hearst patterns

- ▶ “The bow lute, such as the Bambara ndang, is plucked and has an individual curved neck for each string.”
  - ▶ What is a Bambara ndang?
  - ▶ Can answer without knowing what a bow lute is!
- ▶ Source pattern: NP0 such as {NP1, NP2 ... , (and|or)}  
NPn
- ▶ Target inference: for all  $NP_i, i = 1, \dots, n$ , hyponym( $NP_i, NP_0$ )

## More patterns

- ▶ such NP as NP, \* (or|and) NP
    - ▶ "... works by such authors as Herrick, Goldsmith, and Shakespeare."
    - ▶ Lets us infer hyponym("author", "Herrick"), hyponym("author", "Goldsmith"), hyponym("author", "Shakespeare")
  - ▶ NP , NP\* , (or|and) other NP
  - ▶ NP , (especially|including) NP ,\*(or|and) NP
  - ▶ NP1 is a NP2
- 
- ▶ Multiple patterns, each has a recall-precision tradeoff
  - ▶ For each pattern, collect count of matches in a corpus, and quantities derived from raw counts (like PMI)
  - ▶ These can be used as features to train a classifier to predict if NP1 is a hyponym of NP2

## Caveats

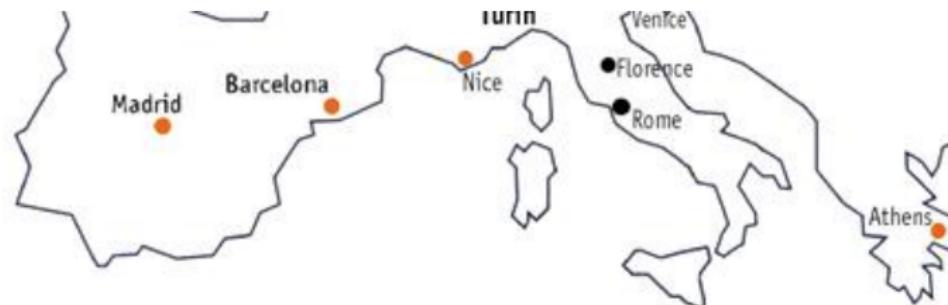
- ▶ Garth Brooks is a country
- ▶ gift such as wall
- ▶ person like Paris

## Caveats

- ▶ Garth Brooks is a country singer
  - ▶ gift such as wall clock
  - ▶ person like Paris Hilton
- 
- ▶ Need noun phrase chunking, e.g. “country singer” or “wall clock”
  - ▶ Also need disambiguation, because “Hilton” may not be mentioned explicitly
  - ▶ Cannot depend on simple phrase queries on a text index

## List/table extraction

airport  
because it is near the city center. For the intercontinental arrivals, the airport offers direct flights to the main European Hubs:



*Direct Flights from/to Torino - Caselle International Airport*

City	Flight hours	Company	Frequency	Fare (Euro)
Amsterdam	2.00	KLM	Daily	350/670
Barcelona	2.00	Iberia	Daily	300/860
Brussels	1.40	Sabena	Daily	300/750
Dusseldorf	2.00	Lufthansa	Daily	320/810
Frankfurt	1.15	Lufthansa	Daily	320/760
Lisbon	3.30	Air Portugal	No Saturday	390/1050
London	2.00	Ryan Air	Daily	Only on line
Madrid	2.10	Iberia	Daily	300/630
Munich	2.00	Lufthansa	Daily	300/690
Paris	1.25	Air France	Daily	150/350

# KnowItAll

- ▶ Specific binary relation: is-a
- ▶ Use Hearst patterns to get high-confidence facts
- ▶ Pick random subsets (say, of size 3–4)
- ▶ Ask Web query with subset
- ▶ Locate HTML lists mentioning **most** elements of subset
- ▶ **Judge** if list is of entities of same type (how?)
- ▶ Extract other candidate entities from list
- ▶ Use **global** stats to validate candidates

# Extractions using Context Patterns

## Induced Patterns (containing sequence "watch")

gold -<ENT>- watch  
diamond -<ENT>- watch  
fake -<ENT>- watches  
bought -<ENT>- watch  
encrusted -<ENT>- watch  
stole -<ENT>- watch  
Richemont AG , -<ENT>- watches  
Rolex and -<ENT>- watches  
buy -<ENT>- watches  
Cartier and -<ENT>- watches  
buy -<ENT>- watch  
gold -<ENT>- watches

Richemont , -<ENT>- watches  
bought -<ENT>- watches  
fake -<ENT>- watch  
diamond -<ENT>- watches  
stole -<ENT>- watches  
buy a -<ENT>- watch  
jewelry , including -<ENT>- watch  
watchmaker -<ENT>- .  
jewelry , including -<ENT>- watches  
stole a -<ENT>- watch  
Rolex watches and -<ENT>- .  
watchmaker -<ENT>- Group

Rolex watches are sold through official -<ENT>- and  
bought a -<ENT>- watch  
watchmaker -<ENT>- SA  
Ulysse -<ENT>- watches  
Rolex watches and -<ENT>- watch  
Rolex , -<ENT>- watch  
Rolex and -<ENT>- watch  
diamond - studded -<ENT>- watch  
diamond - encrusted -<ENT>- watch  
Cartier , and -<ENT>- watches  
buy a -<ENT>- watches  
bought a -<ENT>- watches

## Entities Extracted by Above Patterns (ranked)

Rolex ( <i>most confident</i> )	Fossil	Swatch
Cartier	Tag Heuer	Super Bowl
Swiss	Chanel	SPOT
Movado	Tiffany	Sekonda
Seiko	TechnoMarine	Rolexes
Gucci	Franck Muller	Harry Winston
Patek Philippe	Versace	Hampton Spirit
Piaget	Raymond Weil	Girard Perregaux
Omega	Guess	Frank Mueller
Citizen	Croton	David Yurman
Armani	Audemars Piguet	Chopard
DVD	DVDs	Chinese
Breitling	Montres Rolex	Armitron
Tourneau	CD	NFL ( <i>least confident</i> )

## Extracted Lists Improve NER Taggers

Training Data (Tokens)	Test-a		
	No List	Seed List	Unsup. List
9229	68.27	70.93	<b>72.26</b>
204657	89.52	84.30	<b>90.48</b>

# SEAL: Set Expansion using the Web

[Wang and Cohen, ICDM 2007]

1. Canon
2. Nikon
3. Olympus

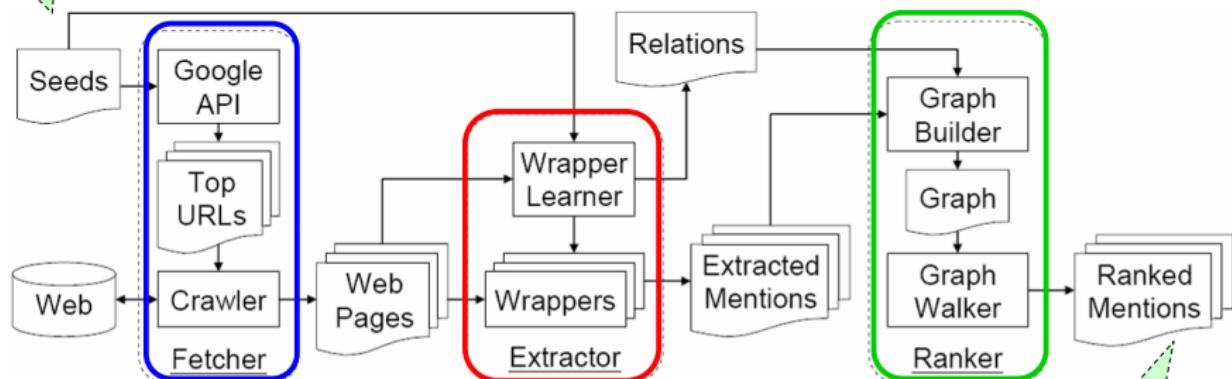


Figure 1. Flow chart of the SEAL system

- **Fetcher**: download web pages from the Web
- **Extractor**: learn wrappers from web pages
- **Ranker**: rank entities extracted by wrappers

4. Pentax
5. Sony
6. Kodak
7. Minolta
8. Panasonic
9. Casio
10. Leica
11. Fuji
12. Samsung
13. ...

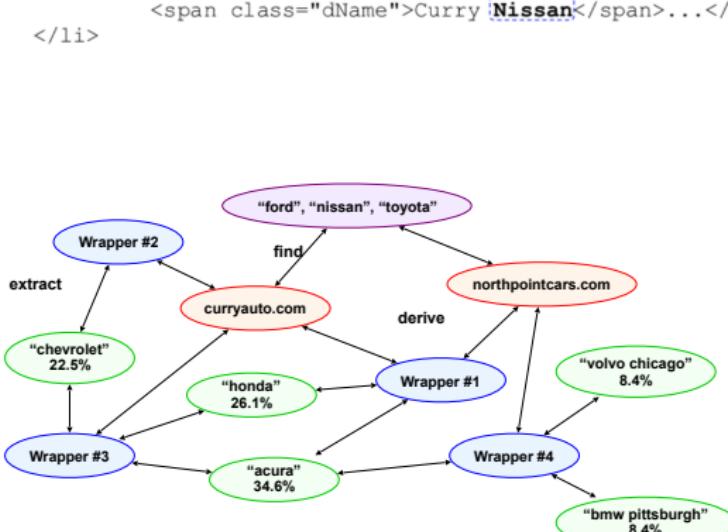
```

<li class="ford"><a href="http://www.curryauto.com/">
</a>
<ul><li class="last"><a href="http://www.curryauto.com/">
    <span class="dName">Curry Ford</span>...</li></ul>
</li>

<li class="nissan"><a href="http://www.curryauto.com/">
</a>
<ul><li class="last"><a href="http://www.geisauto.com/">
    <span class="dName">Curry Nissan</span>...</li></ul>
</li>

```

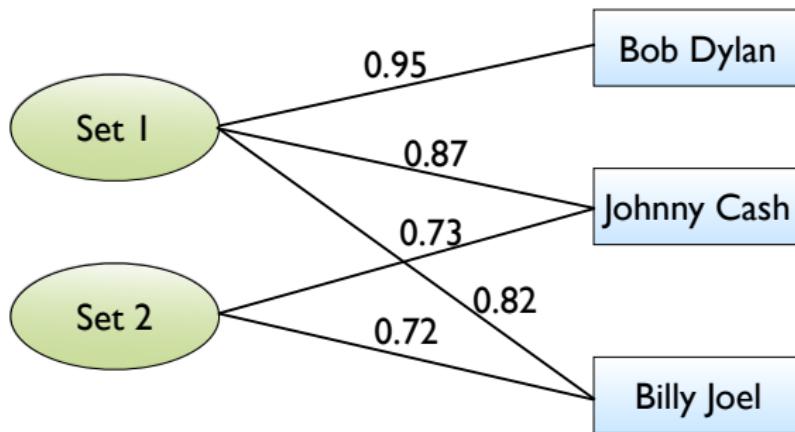
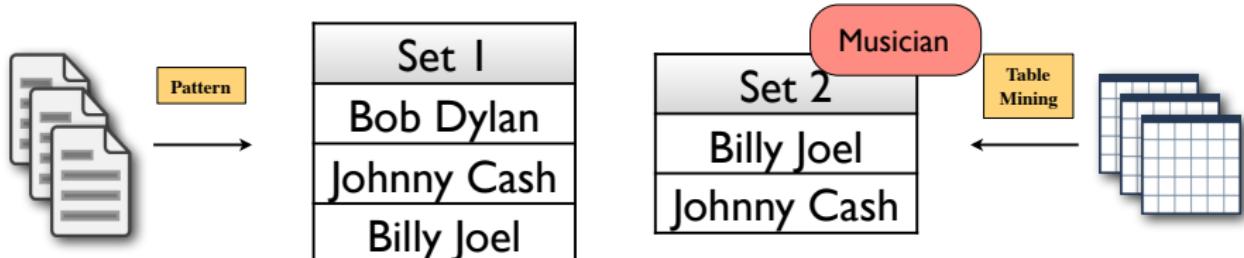
Top three are  
the seeds



#	Entity
1	andrew mccallum
2	michael collins
3	john lafferty
4	naftali tishby
5	fernando pereira
6	zoubin ghahramani
7	daphne koller
8	thomas hofmann
9	thorsten joachims
10	david heckerman
11	nir friedman
12	torn mitchell
13	dan roth
14	william w. cohen
15	mark craven
16	roni rosenfeld
17	david mcalester
18	yoram singer
19	michael i. jordan
20	eugene charniak
21	amir globerson
22	yiming yang
23	yoshua bengio
24	sridhar mahadevan

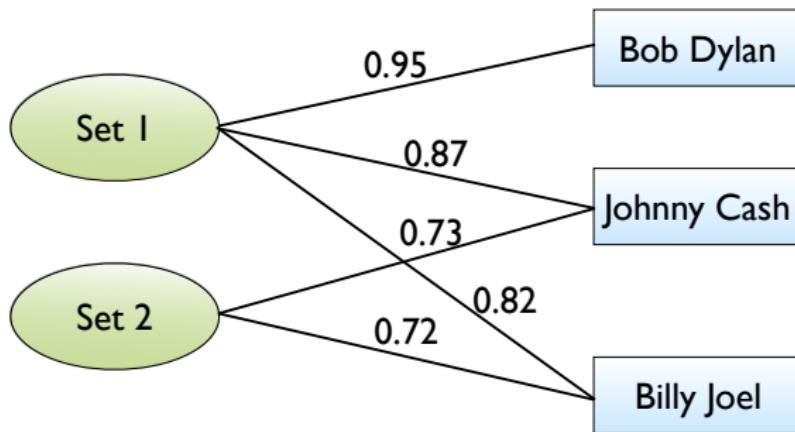
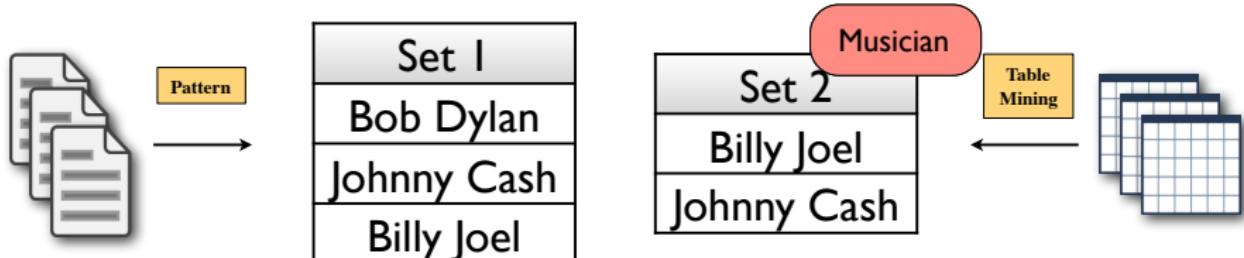
# Combining Patterns and Tables using Graph-based SSL

[Talukdar et al., EMNLP 2008, 2010]



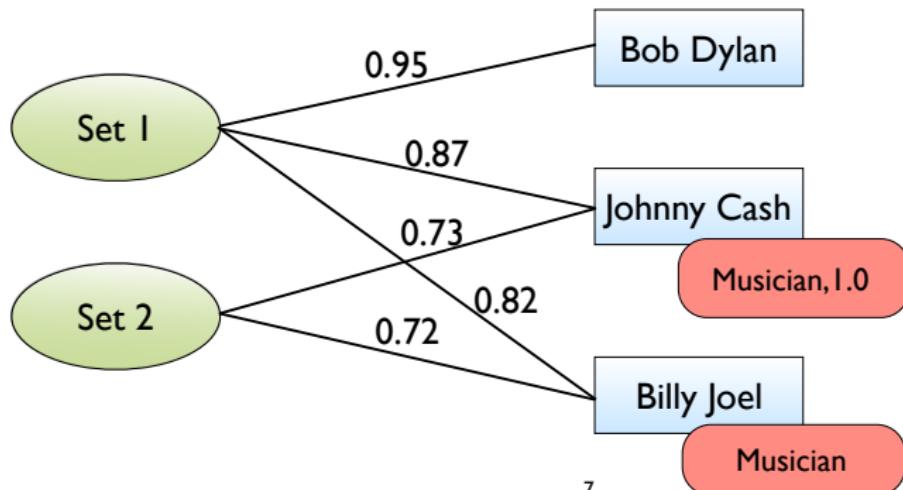
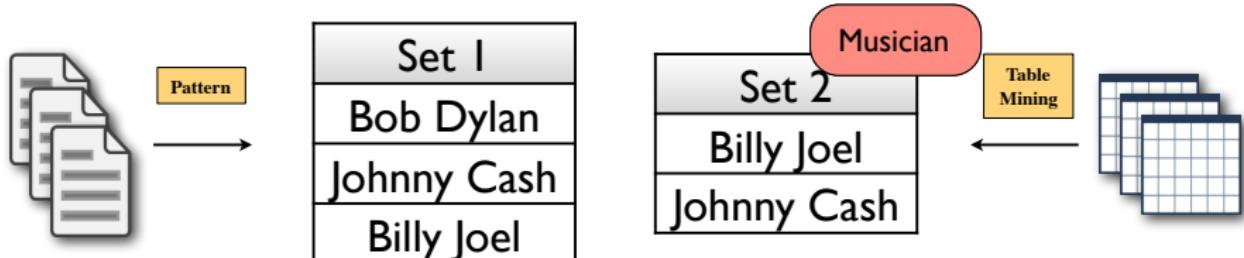
# Combining Patterns and Tables using Graph-based SSL

[Talukdar et al., EMNLP 2008, 2010]



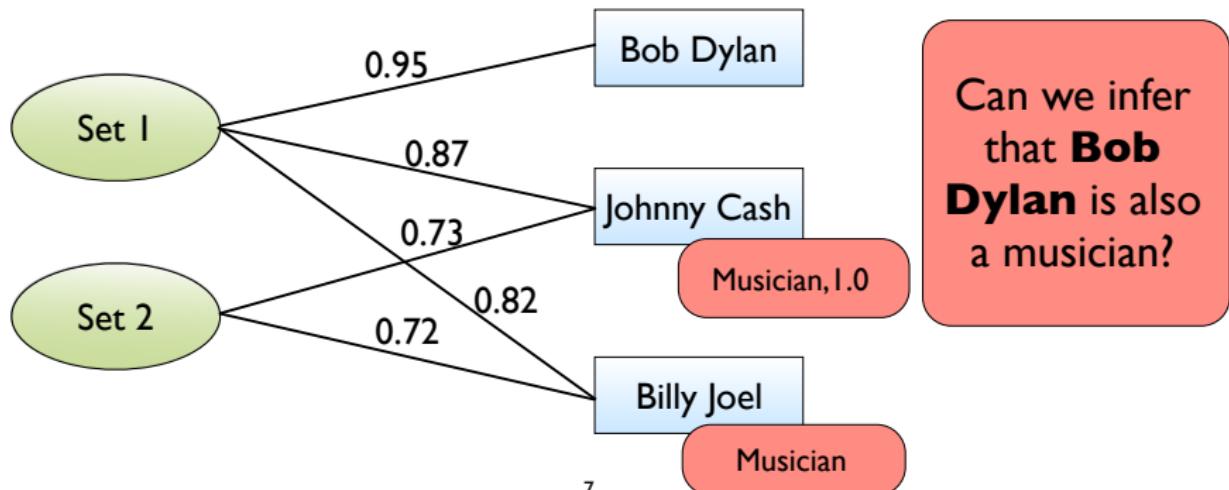
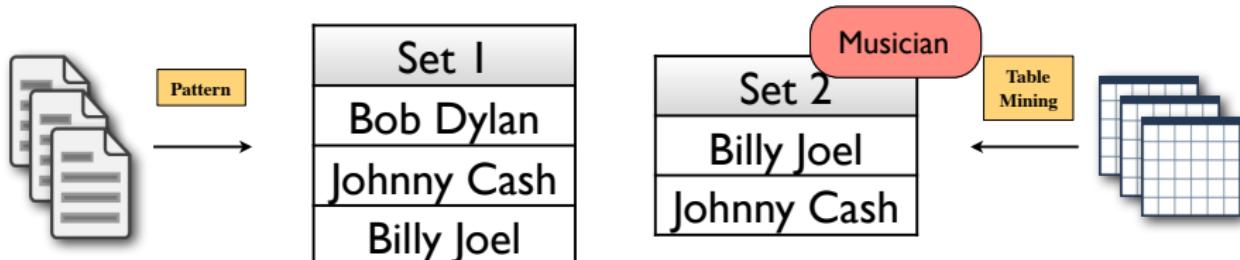
# Combining Patterns and Tables using Graph-based SSL

[Talukdar et al., EMNLP 2008, 2010]



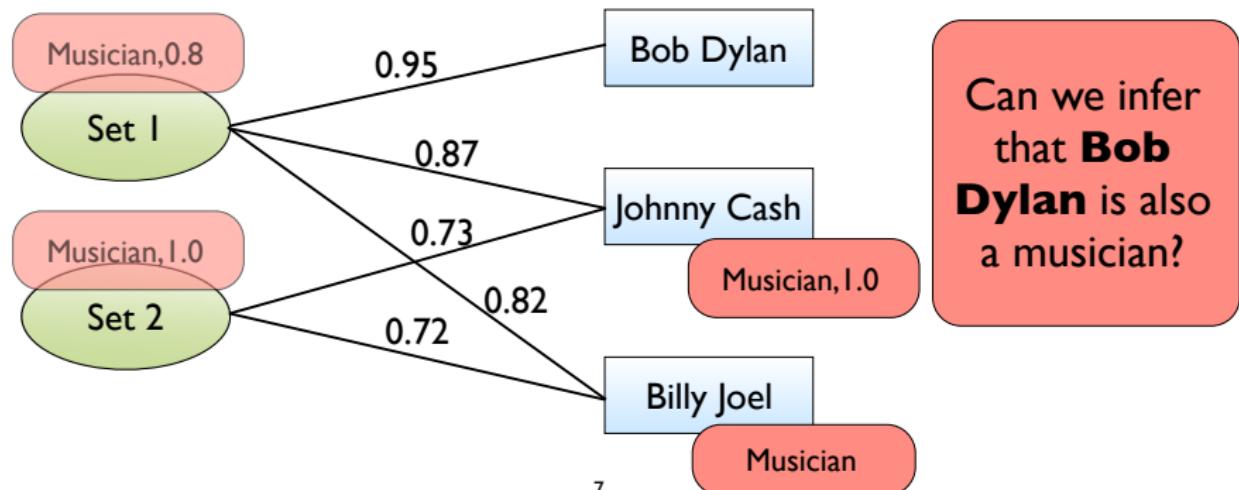
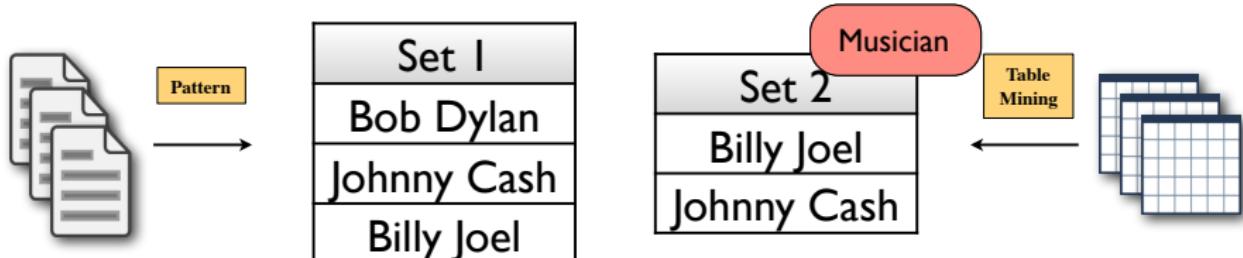
# Combining Patterns and Tables using Graph-based SSL

[Talukdar et al., EMNLP 2008, 2010]



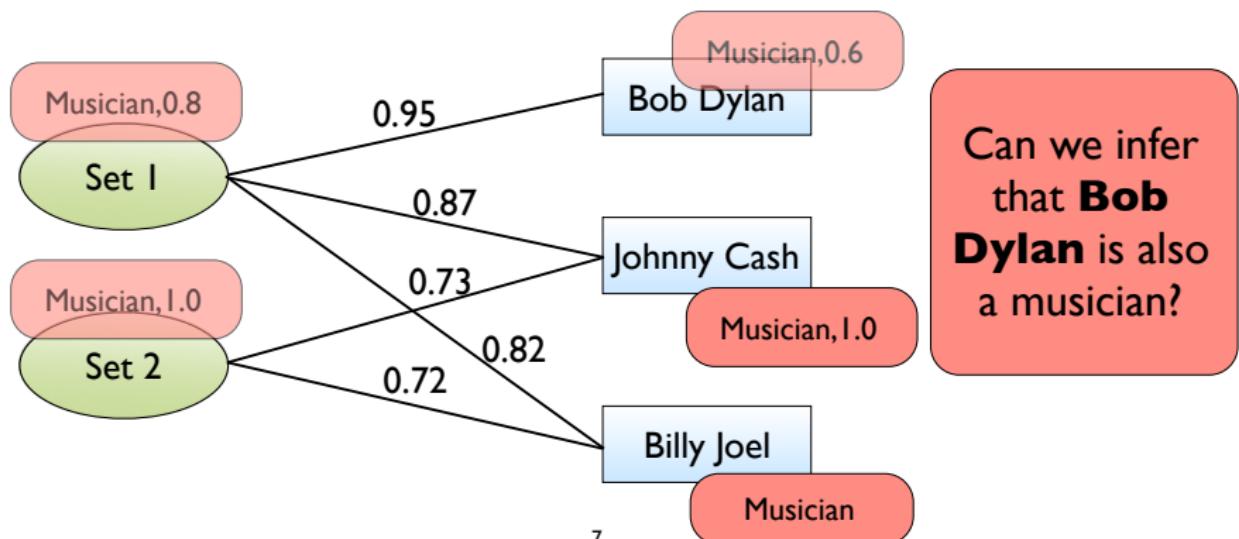
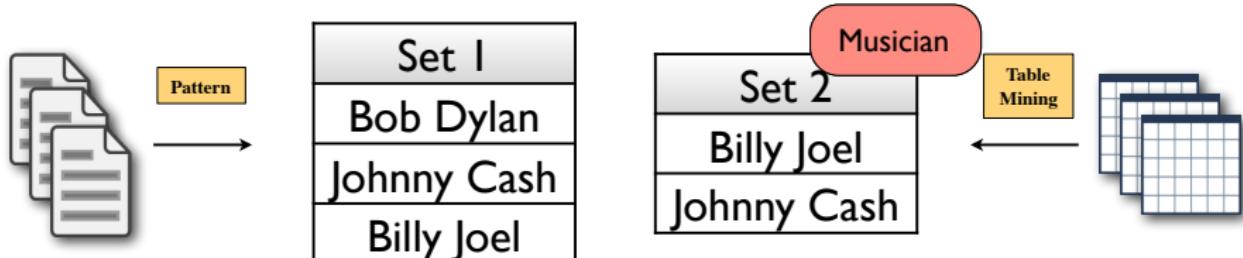
# Combining Patterns and Tables using Graph-based SSL

[Talukdar et al., EMNLP 2008, 2010]



# Combining Patterns and Tables using Graph-based SSL

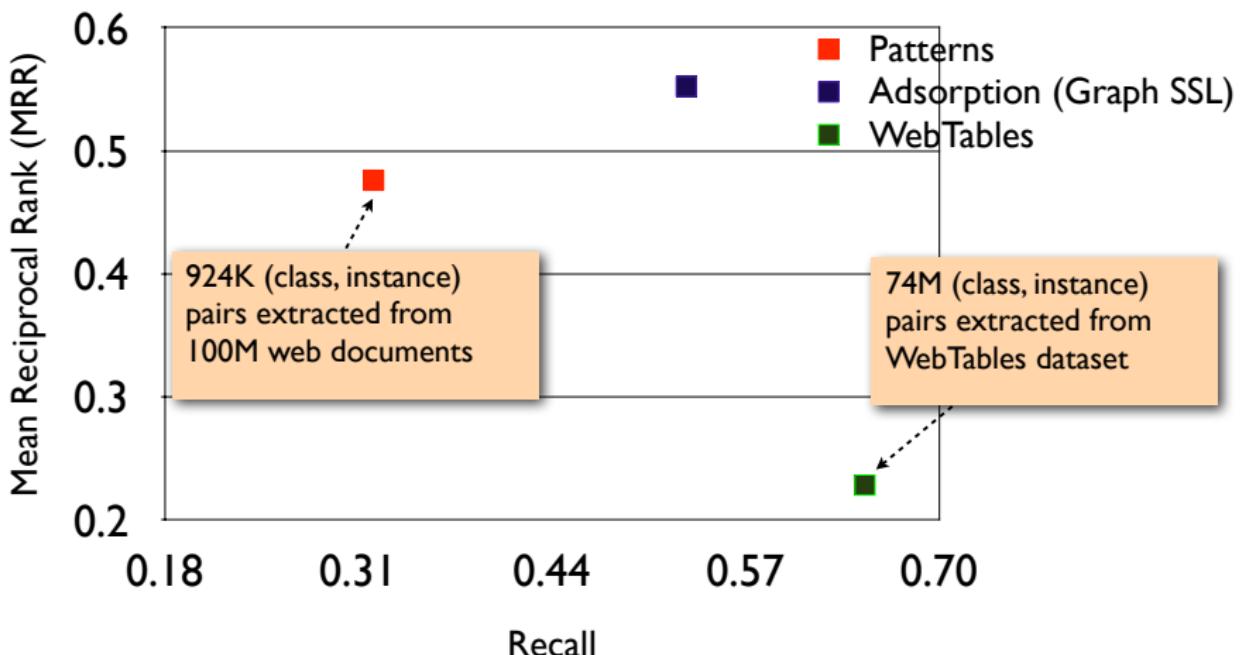
[Talukdar et al., EMNLP 2008, 2010]



# Extraction for Known Instances

Graph with  
1.4m nodes,  
75m edges used.

Evaluation against WordNet Dataset (38  
classes, 8910 instances)

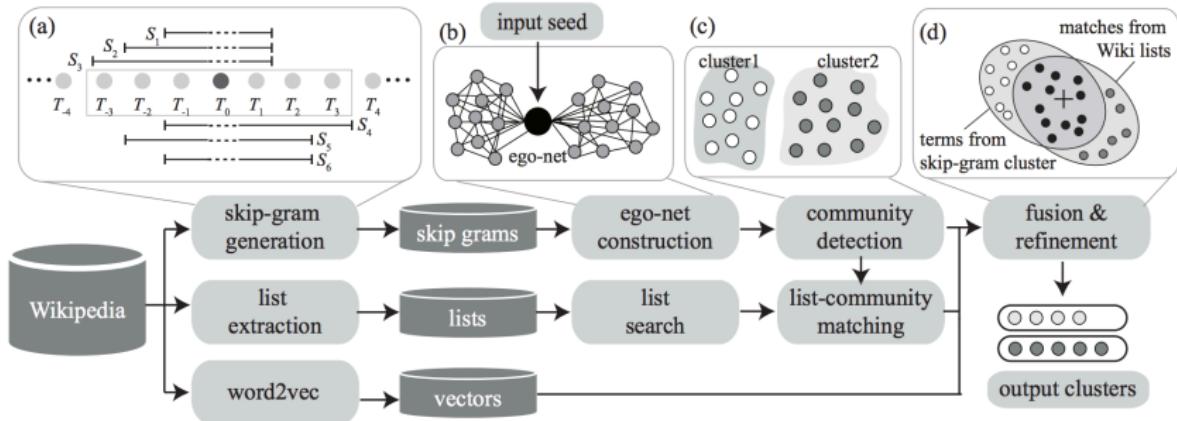


# Extracted Pairs

Total classes: 908 |

Class	Some non-seed Instances found
Scientific Journals	Journal of Physics, Nature, Structural and Molecular Biology, Sciences Sociales et sante, Kidney and Blood Pressure Research, American Journal of Physiology-Cell Physiology, ...
NFL Players	Tony Gonzales, Thabiti Davis, Taylor Stubblefield, Ron Dixon, Rodney Hannan, ...
Book Publishers	Small Night Shade Books, House of Ansari Press, Highwater Books, Distributed Art Publishers, Cooper Canyon Press, ...

# EgoSet [Rong et al., WSDM 2016]



## References

- [1] X. Rong, Z. Chen, Q. Mei, and E. Adar, "Egoset: Exploiting word ego-networks and user-generated ontology for multifaceted set expansion," in *WSDM Conference*, 2016, pp. 645–654.
- [2] S. Brin, "Extracting patterns and relations from the World Wide Web," in *WebDB Workshop*, ser. LNCS, P. Atzeni, A. O. Mendelzon, and G. Mecca, Eds., vol. 1590. Valencia, Spain: Springer, Mar. 1998, pp. 172–183. [Online]. Available: <http://ilpubs.stanford.edu:8090/421/1/1999-65.pdf>
- [3] E. Agichtein and L. Gravano, "Snowball: Extracting relations from large plain-text collections," in *ICDL*, 2000, pp. 85–94. [Online]. Available: <http://www.academia.edu/download/31007490/cucs-033-99.pdf>
- [4] M. Thelen and E. Riloff, "A bootstrapping method for learning semantic lexicons using extraction pattern contexts," in *EMNLP Conference*, Philadelphia, 2002, pp. 214–221.
- [5] T. Mitchell *et al.*, "Never-ending learning," in *AAAI Conference*, 2015. [Online]. Available: [http://www.cs.cmu.edu/~tom/pubs/NELL\\_aaai15.pdf](http://www.cs.cmu.edu/~tom/pubs/NELL_aaai15.pdf)

## References (2)

- [6] E. Riloff, R. Jones *et al.*, "Learning dictionaries for information extraction by multi-level bootstrapping," in *AAAI/IAAI*, 1999, pp. 474–479.
- [7] O. Etzioni, M. Cafarella *et al.*, "Web-scale information extraction in KnowItAll," in *WWW Conference*. New York: ACM, 2004. [Online]. Available: <http://www.cs.washington.edu/research/knowitall/papers/www-paper.pdf>
- [8] R. C. Wang and W. W. Cohen, "Language-independent set expansion of named entities using the web," in *ICDM*, 2007, pp. 342–350.
- [9] P. P. Talukdar and F. Pereira, "Experiments in graph-based semi-supervised learning methods for class-instance acquisition," in *ACL Conference*, 2010, pp. 1473–1481.
- [10] P. P. Talukdar, J. Reisinger, M. Paşa, D. Ravichandran, R. Bhagat, and F. Pereira, "Weakly-supervised acquisition of labeled class instances using graph random walks," in *EMNLP Conference*, 2008, pp. 582–590.

## References (3)

- [11] P. P. Talukdar, T. Brants, M. Liberman, and F. Pereira, "A context pattern induction method for named entity extraction," in *CoNLL*, 2006, pp. 141–148.

# Relation extraction

(Includes slides from Partha Talukdar)

# Introduction: Relation Extraction

**Definition:** A relation is defined in the form of  $r(e_1, e_2)$ , where the  $e_i$  are entities in a predefined relation  $r$ . Easily extended to n-ary relations (events).

## Examples

- ▶ isAcquiredBy relationship between pairs of companies, e.g.,  
 $\text{isAcquiredby(Google, YouTube)}$
- ▶ isAppointedCeoOf relationship between a person and company
- ▶ geneCausesDisease between gene and disease

## Types

- ▶  $?(e_1, e_2)$
- ▶  $r(e_1, ?)$
- ▶  $r(?, ?)$
- ▶ Macro (corpus-level) vs Micro (sentence-level)

## Example relations

- ▶ “is acquired by” relationship between pairs of companies
  - ▶ “is appointed CEO of” relationship between a person and company,
  - ▶ “is employee of” relationship between a person and an organization
  - ▶ ACE Task
    - ▶ “located at”
    - ▶ “near”,
    - ▶ “part”,
    - ▶ “role”,
    - ▶ “social”
- over pairs from five top-level entity types “person”,  
“organization”, “facility”, “location”, and, “geo-political entity”.

## Example relations (2)

- ▶ BioCreAtIvE II Protein-Protein Interaction

1. gene-disease relations,
2. protein-protein interaction, and
3. subcellular regularizations

**Closed domain** relation extraction means, for each relation type, we collect some training examples. I.e., human effort scales with the number of distinct relation types we want to recognize.

# Dominant Approaches to Relationship Extraction

- ▶ **Supervised**
  - ▶ For each relation type, we collect annotated sentences as training examples.
  - ▶ Preferred approach if the types of relations of interest is a small set
  - ▶ Pros: Several effective methods: CRF, LSTM, Bi-LSTM etc. Works quite well when enough training data is available.
  - ▶ Cons: Human effort required scales with the number of distinct relation types. Not feasible at scale.
- ▶ **Weakly-supervised**
  - ▶ Supervision is provided at the relation instance level, not annotated sentences (next slide).
  - ▶ Pros: Supervision size is small and easy to provide. Much more scalable and practical.
  - ▶ Cons: There is more noise in the resulting training data, results in a more challenging learning problem.
  - ▶ State-of-the-art is at around 40% Precision at 30% recall → lot of headroom for improvement

## (Weak) supervision setup

Given a corpus  $D$ , set of relationship types  $r_1, \dots, r_k$ , entity types  $T_{r1}, T_{r2}$  forming arguments of relationship type  $r$ , and a seed set  $S$  of examples of the form  $(E_{i1}, E_{i2}, r_i)$   $1 \leq i \leq N$  indicating that  $E_{i1}$  has relationship  $r_i$  with  $E_{i2}$ .

$E_1$	$E_2$	$r$	Label
Alon Halevy	Anhai Doan	IsPhDAdvisorOf	+
Donald Knuth	Andrei Broder	IsPhDAdvisorOf	+
Jeff Ullman	Surajit Chaudhari	IsPhDAdvisorOf	+
Alon Halevy	Dan Suciu	IsPhDAdvisorOf	-
Google	YouTube	Acquired	+
Google	Yahoo!	Acquired	-
Microsoft	Powerset	Acquired	+

## Sources of relation extraction features

- ▶ Surface tokens and their shapes
- ▶ Part of speech and chunk tags
- ▶ Constituency and dependency parses
- ▶ Word, type, relation embeddings (later)

## Surface Tokens

The tokens around and in-between the two entities often hold strong clues for relationship extraction.

*<Company> Kosmix </Company> is located in the <Location> Bay Area </Location>*

A “is\_situated” relationship between a Company entity and Location entity indicated by token “located” and bigram tokens “located in”

*... the Center for Disease Control and Prevention, which is in the front line of the world's response to the deadly <Disease> Ebola </Disease> epidemic in <Location> Zaire </Location>,*

A “outbreak” relationship between a disease and location is indicated by keyword “epidemic”.

## Part of speech tags

Verbs in a sentence are key to defining the relationship between entities, that are typically nouns or noun phrases.

*⟨Location⟩ The University of Helsinki ⟨/Location⟩ hosts  
⟨Conference⟩ ICML ⟨/Conference⟩ this year.*

Word “hosts” as a verb is a clue..

*The/DT University/NNP of/IN Helsinki/NNP hosts/VBZ  
ICML/NNP this/DT year/NN*

## Syntactic parse tree structure

*<Location> Haifa </Location>, located 53 miles from  
<Location> Tel Aviv </Location> will host <Conference> ICML  
</Conference> in 2010.*

This tree brings “ICML” closer to “Haifa” than “Tel Aviv” because “Haifa” is the head of the noun phrase “Haifa, located 53 miles from Tel Aviv” which forms the subject of the verb phrase “will host ICML in 2010”.

# Syntactic parse tree structure

```
(ROOT
  (S
    (NP
      (NP (NNP Haifa))
      (VP (VBN located)
        (PP
          (NP (CD 53) (NNS miles))
          (IN from)
          (NP (NNP Tel) (NNP Aviv))))))
    (VP (MD will)
      (VP (VB host)
        (NP
          (NP (NNP ICML))
          (PP (IN in)
            (NP (CD 2010))))))))
```

# Dependency graph

Links each word to the words that depend on it

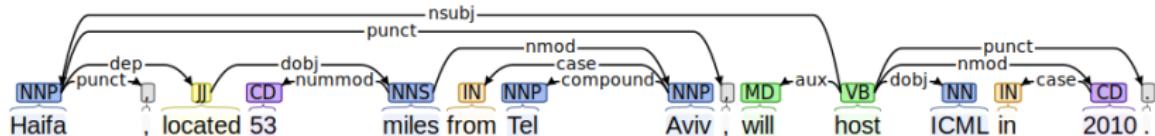
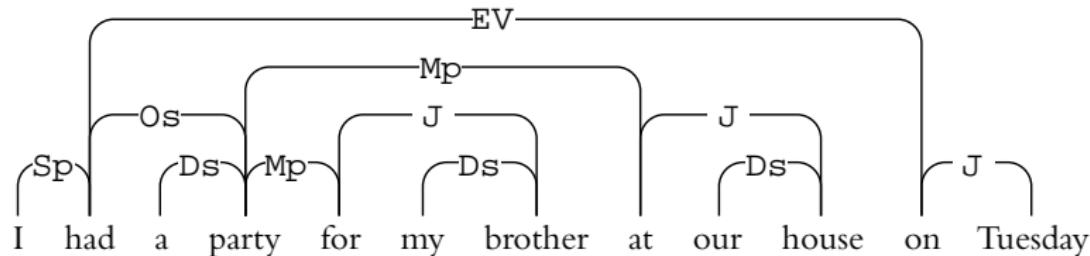


Figure: Dependency parse of a sentence.

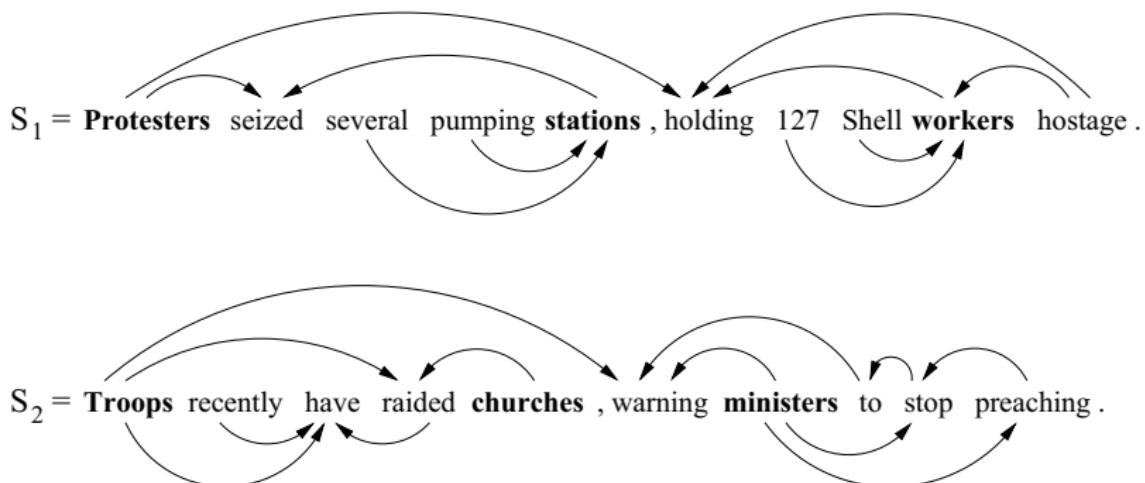
- ▶ Verb “host” is linked to both “Haifa” a location entity and with “ICML” a conference entity and this directly establishes a close connection between them
- ▶ In contrast, the path between ICML and Tel Aviv goes through “Haifa” and “Located”.

# Dependency parsing



- ▶ Links tokens in sentence
- ▶ Edges encode purpose of syntax
- ▶ Often used for [slot filling](#) :
  - ▶ Which company bought out which company?
  - ▶ For how much money?
  - ▶ How much in stocks vs. cash?
  - ▶ Where did the accident happen?
  - ▶ How many were injured?

## Dependency graph examples



- ▶ Observe similarity in graph structure despite no word-based similarity
- ▶ Essential to link multiple agents in different relationships within same sentence

# Path kernel

Relation Instance	Shortest Path in Undirected Dependency Graph
$S_1$ : protesters AT stations	<b>protesters</b> → seized ← <b>stations</b>
$S_1$ : workers AT stations	<b>workers</b> → holding ← protesters → seized ← <b>stations</b>
$S_2$ : troops AT churches	<b>troops</b> → raided ← <b>churches</b>
$S_2$ : ministers AT churches	<b>ministers</b> → warning ← troops → raided ← <b>churches</b>

- ▶ Hypothesis: (Undirected) path connecting entities  $e_1$  and  $e_2$  in dependency graph largely determines if relation  $R(e_1, e_2)$  holds
- ▶ (Also need type info  $e_1 \in T_1 = \text{person}$ ,  $e_2 \in T_2 = \text{location}$ )
- ▶ To generalize beyond specific words, coarsen path representation into word categories

$$\begin{bmatrix} \text{protesters} \\ \text{NNS} \\ \text{Noun} \\ \text{Person} \end{bmatrix} \times [\rightarrow] \times \begin{bmatrix} \text{seized} \\ \text{VBD} \\ \text{Verb} \end{bmatrix} \times [\leftarrow] \times \begin{bmatrix} \text{stations} \\ \text{NNS} \\ \text{Noun} \\ \text{Facility} \end{bmatrix}$$

## Path kernel (2)

- ▶ How about “He never went to Paris”
- ▶ Next: need to define similarity (kernel)  $K(\vec{x}, \vec{y})$  between two such paths
  - ▶ **his** → actions ← in ← **Brcko**
  - ▶ **his** → arrival ← in ← **Beijing**
- ▶ Corresponding sequences of sets are
  - ▶  $\vec{x} = (x_1, \dots, x_7)$  where  $x_1 = \{\text{his, PRP, Person}\}$ ,  $x_2 = \{\rightarrow\}$ ,  $x_3 = \{\text{actions, NNS, Noun}\}$ ,  $x_4 = \{\leftarrow\}$ ,  $x_5 = \{\text{in, IN}\}$ ,  $x_6 = \{\leftarrow\}$ ,  $x_7 = \{\text{Brcko, NNP, Noun, Location}\}$
  - ▶  $\vec{y} = (y_1, \dots, y_7)$  where  $y_1 = \{\text{his, PRP, Person}\}$ ,  $y_2 = \{\rightarrow\}$ ,  $y_3 = \{\text{arrival, NNS, Noun}\}$ ,  $y_4 = \{\leftarrow\}$ ,  $y_5 = \{\text{in, IN}\}$ ,  $y_6 = \{\leftarrow\}$ ,  $y_7 = \{\text{Beijing, NNP, Noun, Location}\}$
- ▶ For starters, no stretching or compression:  $K(\vec{x}, \vec{y}) = 0$  unless the sequences have the same number of sets in them; i.e., if  $\vec{x} = (x_1, \dots, x_m)$  and  $\vec{y} = (y_1, \dots, y_n)$ ,  $K(x, y) > 0$  only if  $m = n$

## Path kernel (3)

- ▶ In which case,

$$K((x_1, \dots, x_n), (y_1, \dots, y_n)) = \prod_{i=1}^n |x_i \cap y_i|$$

- ▶ In the example,  $K(\vec{x}, \vec{y}) = 3 \times 1 \times 1 \times 1 \times 2 \times 1 \times 3 = 18$
- ▶ Now use standard SVM classifier
- ▶ Works well for NIST's ACE-style tasks
  - ▶ Entity types are Person, Organization, Facility, Location, and GeopoliticalEntity
  - ▶ Relation types are Role, Part, Located, Near, and Social
  - ▶ 7646 intra-sentential relations
  - ▶ Precision 66–72%, recall 39–44%,  $F_1$  51–53%

## Weak Supervision Method Type 1: Bootstrapping

- ▶ Start with a small table of known facts

Isaac Asimov	The Robots of Dawn
--------------	--------------------

David Brin	Startide Rising
------------	-----------------

James Gleick	Chaos: Making a New Science
--------------	-----------------------------

Charles Dickens	Great Expectations
-----------------	--------------------

William Shakespeare	The Comedy of Errors
---------------------	----------------------

- ▶ Find mentions of known authors and books in the corpus:

*The **Robots of Dawn** is a “whodunit” science fiction novel by Isaac Asimov, first published in 1983. It is part of Asimov’s Robot series.*

- ▶ Induce and evaluate patterns on known data

- ▶ \*prefix, author, middle, title, suffix\*
- ▶ <LI><B>title</B> by author (
- ▶ <i>title</i> by author (

- ▶ Find matches to patterns over corpus (scan/index?)

## Weak Supervision Method Type 1: Bootstrapping (2)

- ▶ Import confident extractions into database
- ▶ Rinse and repeat
  - 1: input seed tuples  $\{(e_{i1}, e_{i2}), i = 1, \dots, n\}$
  - 2: **while** database not big enough **do**
    - 3: find snippets in corpus where seed tuples are mentioned
    - 4: tag entities in snippets
    - 5: generate new **patterns**  $L, t_1, C, t_2, R$  or  $L, t_2, C, t_1, R$
    - 6: apply new patterns over whole corpus
    - 7: import newly extracted tuples into database
- ▶ Brin bootstrapped as follows: 5 facts  $\rightarrow$  199 occurrences  $\rightarrow$  3 patterns  $\rightarrow$  4047 proposed facts  $\rightarrow$  105 more patterns  $\rightarrow$  9369 proposed facts
- ▶ Quality control needed
  - ▶ Which extracted tuples are likely to be correct?
  - ▶ Which patterns are sufficiently reliable and useful?

## Weak Supervision Method Type 1: Bootstrapping (3)

- ▶ Use some notion like precision or  $F_1$  as pattern confidence  $\text{Conf}(P)$ , e.g.,

$$\text{Conf}(P) = \frac{n^+}{n^+ + n^-},$$

where  $n^+$  ( $n^-$ ) is the number of known true (false) statements matched by  $P$

## More pattern examples

Organization $e_1$	Location of headquarters $e_2$
Microsoft	Redmond
Exxon	Irving
IBM	Armonk
Boeing	Seattle
Intel	Santa Clara

- ▶  $e_1$ 's headquarters in  $e_2$
- ▶  $e_2$ -based  $e_1$
- ▶  $e_1, e_2$

## How to represent patterns

Organization $e_1 \in t_1$	Headquarter location $e_2 \in t_2$
Microsoft	Redmond
Exxon	Irving
IBM	Armonk
Boeing	Seattle
Intel	Santa Clara

- ▶  $L = \{(\text{the}, 0.2)\}$
- ▶  $t_1 = \text{location}$
- ▶  $C = \{(-, 0.5), (\text{based}, 0.5)\}$
- ▶  $t_2 = \text{organization}$
- ▶  $R = \{\}$

## How to induce and evaluate patterns

- ▶ Pattern generated by clustering contexts of known facts
- ▶ Keep a held-out set of verified true and false is-a statements
- ▶ Apply pattern  $P$  on held-out statements
- ▶ Use some notion like precision or  $F_1$  as pattern confidence  $\text{Conf}(P)$ , e.g.,

$$\text{Conf}(P) = \frac{n^+}{n^+ + n^-},$$

where  $n^+$  ( $n^-$ ) is the number of known true (false) statements matched by  $P$

## Pattern match

- ▶ Two patterns  $P_1 = (L_1, a_1, C_1, b_1, R)$  and  $P_2 = (L_2, a_2, C_2, b_2, R_2)$  have nonzero match only if  $a_1 = a_2$  and  $b_1 = b_2$
- ▶ In that case, the match score is

$$\text{Match}(P_1, P_2) = L_1 \cdot L_2 + C_1 \cdot C_2 + R_1 \cdot R_2$$

i.e., sum of dot products of word bags

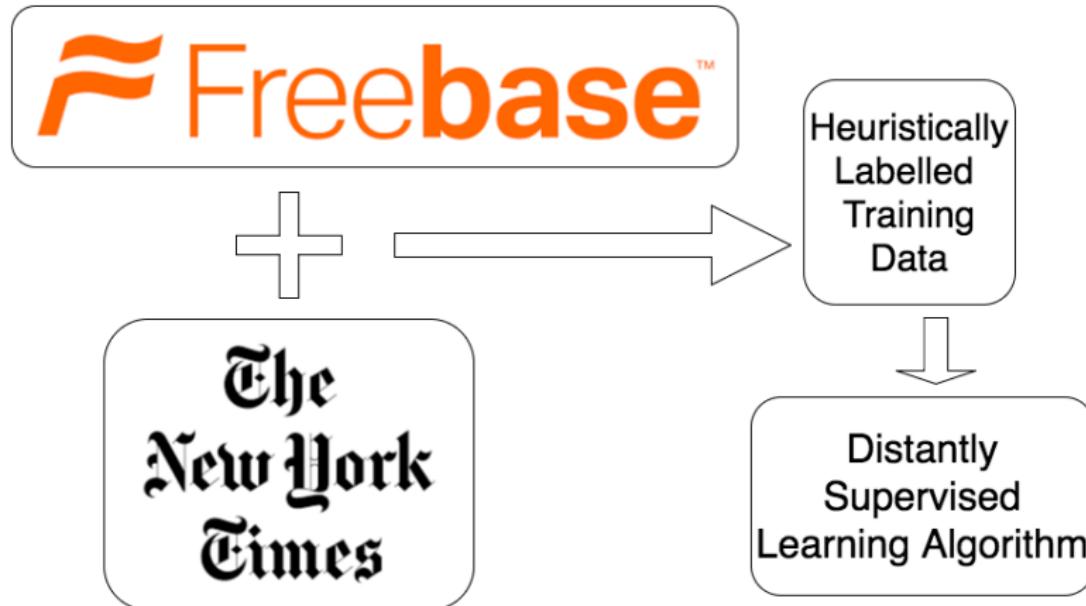
- ▶ Can scale to make it a belief between 0 and 1, or threshold it

## Confidence of match

$$1 - \prod_P (1 - \text{Conf}(P) \text{Match}(P, S))$$

- ▶  $P$  is a pattern
- ▶  $S$  is a snippet containing  $e_1 \in t_1, e_2 \in t_2$
- ▶ A soft-or formulation
- ▶ Can iterate between match and pattern confidence using EM

## Weak Supervision Method Type 2: Distant Supervision



# Relation Extraction using Distant Supervision Example

Entity 1 => Narendra Modi  
Entity 2 => India

Relation => Prime\_Minister\_of

Instance Set =>

- \* Narendra Modi, Prime Minister of India visited USA
- \* Narendra Modi payed respects at Amar Jawan Jyoti on India's Independence day
- \* Narendra Modi represented India at recent BRICS Summit
- \* Narendra Modi, Indian Prime Minister welcomed delegates from UK visiting Delhi

Figure: Example of an Instance Set in Distant Supervision training set

- ▶ Disadvantage: Noisy labelled data
- ▶ Advantage: Large amount of labelled data at very low cost

## Multi-instance multi-label relation extraction

Sentence	Latent Label
Barack Obama is the 44th and current President of the United States.	EmployedBy
Obama was born in the United States just as he has always said.	BornIn
United States President Barack Obama meets with Chinese Vice President Xi Jinping today.	EmployedBy
Obama ran for the United States Senate in 2004.	-

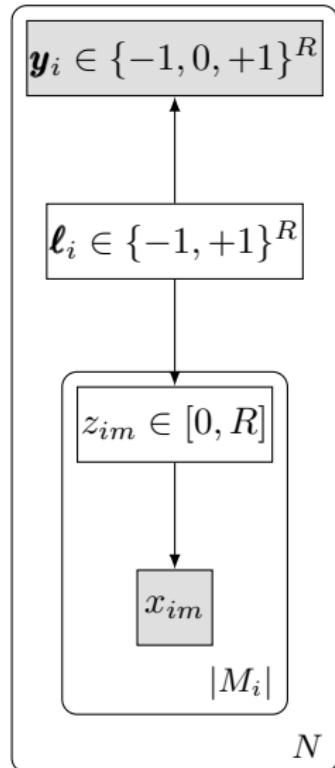
- ▶  $DB = \{ BornIn(\text{Barack Obama}, \text{United States}), EmployedBy(\text{Barack Obama}, \text{United States}) \}$
- ▶ Multiple relations; each sentence may support 0 or 1 relation between two entities mentioned in it

## Multi-instance multi-label relation extraction (2)

- ▶ Multiple relations may hold between entities  $e_1, e_2$ , evidenced in different sentences
- ▶ If a relation  $r$  holds between entities  $e_1, e_2$ , then *at least one* sentence has to support with evidence
- ▶ If relation  $r$  does not hold between entities  $e_1, e_2$ , then there can be no evidence sentence
- ▶ Assuming all sentences are evidence of *some* relation pollutes training data

## Generative story

- ▶  $N$  is the number of entity pairs
- ▶ Entity pair  $q_i = (e_{is}, e_{io})$  has latent label  $\ell_i \in \{-1, +1\}^R$
- ▶  $R$  is the number of relations
- ▶ Latent labels are noisily sampled into observed labels  $y_i \in \{-1, 0, +1\}^R$
- ▶ Using a  $2 \times 3$  probability table
- ▶ 0 means there is no supervised label
- ▶ Entity pair  $q_i$  has mention contexts  $M_i$
- ▶ Context  $(i, m)$  chooses to express 0 or 1 label from  $\ell_i$ , expressed in  $z_{im}$
- ▶ Latent sentence label  $z_{im}$  controls the generation of the observed sentence features  $x_{im}$



## Simplified (conditional) story: MIML-RE

- ▶ Assume completely supervised  $\mathbf{y}_i \in \{-1, +1\}$
- ▶ I.e.,  $\ell_i$  has no role, or collapsed with  $\mathbf{y}_i$
- ▶ During training,  $\mathbf{y}_i$  and  $\mathbf{x}_i = \{x_{im}\}$  known
- ▶ Parameterization:
  - ▶  $\mathbf{w}_z$  used to predict  $z_{im}$  from  $x_{im}$
  - ▶ A regular logistic regression with hand-picked features from  $x_{im}$
  - ▶  $\mathbf{w}_y$  used to combine  $\mathbf{z}_i = \{z_{im}\}$  across mentions  $m$  into  $\hat{\mathbf{y}}_i$  and minimize loss wrt  $\mathbf{y}_i$
  - ▶ Logistic regression with (features derived from)  $\mathbf{z}_i$  as input and each  $y_{ir}$  as output
- ▶ Apart from  $\ell_i$  disappearing, directions of arrows no longer as in generative story
- ▶ Use hard-EM:
  - ▶ During the hard-E step, re-estimate each  $z_{im}$  while holding all other  $z_{im'}$  (and  $\mathbf{w}_z, \mathbf{w}_y$ ) fixed

## Simplified (conditional) story: MIML-RE (2)

- ▶ With re-estimated  $\{\mathbf{z}_i\}$ , quite simple to re-estimate  $\mathbf{w}_z, \mathbf{w}_y$  in a standard M-step
- ▶ For the hard-E step:

$$\begin{aligned} p(z_{im}, \mathbf{y}_i | \mathbf{x}_i; \mathbf{w}_z, \mathbf{w}_y) &\approx p(z_{im} | x_{im}, \mathbf{w}_z) p(\mathbf{y}_i | \mathbf{z}_{i \setminus m}, \mathbf{w}_y) \\ &\approx p(z_{im} | x_{im}; \mathbf{w}_z) \prod_r p(y_{ir} | \mathbf{z}_{i \setminus m}; w_{yr}) \end{aligned}$$

- ▶ Here  $\mathbf{z}_{i \setminus m}$  denotes  $\mathbf{z}_i$  with the  $m$ th element replaced with  $z_{im}$
- ▶ Hard-E step amounts to setting

$$z_{im}^* = \underset{z_{im} \in [0, R]}{\operatorname{argmax}} p(z_{im}, \mathbf{y}_i | \mathbf{x}_i; \mathbf{w}_z, \mathbf{w}_y)$$

## Simplified (conditional) story: MIML-RE (3)

- ▶ For the M step

$$\boldsymbol{w}_z^* \leftarrow \operatorname{argmax}_{\boldsymbol{w}_z} \sum_i \sum_{m \in M_i} \log p(z_{im}^* | x_{im}; \boldsymbol{w}_z)$$

$$\forall r : \quad w_{yr}^* \leftarrow \operatorname{argmax}_{w_{yr}} \log p(y_{ir} | \boldsymbol{z}_{ir}^*; w_{yr})$$

- ▶ Features for  $\boldsymbol{w}_z$  include
  - ▶ Head words and shapes of entity mentions
  - ▶ Dependency path between entity mentions
  - ▶ Sequence of POS tags between entity mentions
- ▶ Features for  $w_{yr}$  include
  - ▶ “At least one” feature:  $\max_m [\![z_{im} = r]\!]$
  - ▶ Correlations between relation labels: feature for  $r' \neq r$  is  $\sum_m [\![z_{im} = r']\!]$  (need to read code to know exactly how these features are defined)

## MultiR

- ▶  $q_i = (e_{i1}, e_{i2}) \in E \times E$  is the  $i$ th entity pair
- ▶ Boolean random variable  $Y_{ir}$  for each relation  $r$ , representing if  $r(e_{i1}, e_{i2})$  is true
- ▶ Sentence  $x_{im} \in M_i$  mentions both entities
- ▶ Random multiway variable  $Z_{im} \in [0, R]$  as before
- ▶ Problems for different  $i$  are isolated except shared model weights; will hide  $i$  when possible

$$p(\mathbf{y}, \mathbf{z} | \mathbf{x}; \theta) \propto \prod_m \phi^{\text{extract}}(z_m, x_m; \theta) \prod_r \phi_r^{\text{join}}(y_r, \mathbf{z})$$

- ▶  $\phi^{\text{join}}$  is a deterministic OR:

$$\phi_r^{\text{join}}(1, \mathbf{z}) = \bigvee_m \llbracket z_m = r \rrbracket$$

$$\phi_r^{\text{join}}(0, \mathbf{z}) = \bigwedge_r \llbracket z_m \neq r \rrbracket$$

## MultiR (2)

- ▶  $\phi^{\text{extract}}$  is a standard log-linear model:

$$\phi^{\text{extract}}(z_m, x_m; \theta) = \exp(\theta \cdot f(z_m, x_m))$$

- ▶ Perceptron update pseudocode:

```
1: while model improves do
2:   for each instance ( $x_i, y_i$ ) do
3:      $y', z' \leftarrow \operatorname{argmax}_{y, z} p(y, z | x_i; \theta)$ 
4:     if  $y_i \neq y'$  then
5:        $z^* \leftarrow \operatorname{argmax}_z p(z | x_i, y_i; \theta)$ 
6:        $\theta \leftarrow \theta + f(x_i, z^*) - f(x_i, z')$ 
```

- ▶ How to find  $\operatorname{argmax}_z p(z | x_i, y_i; \theta)$ ?
- ▶ I.e., given mentions *and* global label constraints, how to assign labels  $z_{im}$  to individual mentions?

## MultiR (3)

- ▶ Construct a bipartite graph for instance  $i$ :
  - ▶ On the left, one node for each mention sentence  $m$
  - ▶ On the right, one node for each relation  $r$  where  $y_{ir} = 1$
  - ▶ Edge weight between  $m$  and  $r$  is  $\phi^{\text{extract}}(x_{im}, r; \theta)$
- ▶ Select a subset of edges such that
  - ▶ Sum of edge weights is maximized
  - ▶ Each node  $m$  is incident on exactly (at most?) one selected edge
  - ▶ Each node  $r$  is incident on at least one selected edge
- ▶ First compute maximum weighted bipartite matching
- ▶ Then select edges for nodes not incident on any selected edge
- ▶ Takes  $\sim (R + |M_i|)[R|M_i| + (R + |M_i|)\log(R + |M_i|)]$  time

## Back to the semisupervised scenario

- ▶ I.e., with latent  $\ell_i$  and observed  $y_i$  (during training)
- ▶ Parameterization changes from  $w_z, w_y$  in MIML-RE to  $w_z, w_\ell$
- ▶ The connection between  $\ell$  and  $y$  is enforced by the 6-cell table  $p(y|\ell)$ ; looks like this was not trained

	$y \rightarrow$	-1	0	+1
$\ell \downarrow$	-1	0	1	0
	+1	0	$1/2$	$1/2$

- ▶ During inference, no point using this table to predict 0, so

$$z_{im}^* = \underset{z \in [0, R]}{\operatorname{argmax}} p(z|x_{im}; \mathbf{w}_z)$$

$$y_{ir}^* = \underset{y \in \{-1, 1\}}{\operatorname{argmax}} p(y|z_i^*; \mathbf{w}_\ell)$$

- ▶ During training, we now have latent variables  $\ell$  in addition to  $z$ ; for both use hard-E step

## Back to the semisupervised scenario (2)

- ▶ Will greedily fix  $\ell$  first by snapping  $y = 0$  cases to  $\pm 1$ , then fix  $z$
- ▶ Per-instance average number of “on” relations:

$$\mu = \frac{1}{N} \sum_{i=1}^N \sum_{r=1}^R \mathbb{I}[y_{ir} = +1]$$

- ▶ Need to stabilize  $\ell$  to stay close to this average
- ▶ Find  $\ell$  to approximately maximize

$$\log p(\ell | \mathbf{y}, \mathbf{x}; \mathbf{w}_z, \mathbf{w}_\ell)$$

$$= \text{const.} + \log p(\mathbf{y} | \ell) - \text{large} \left[ \sum_r \mathbb{I}[\ell_r = +1] - \mu \right]^2$$

$$+ \log p(\ell | \mathbf{x}; \mathbf{w}_z, \mathbf{w}_\ell)$$

## Back to the semisupervised scenario (3)

- 1:  $\ell \leftarrow \mathbf{y}$
- 2: **if**  $\ell_r = 0$  **then**
- 3:     Replace with  $\ell_r \leftarrow -1$
- 4: Let  $R_+ = \{r : \ell_r = +1\}$  and  $R_- = \{r : \ell_r = -1\}$
- 5: **while**  $|R_+| < \mu$  **do**     /\*  $|R_+| > \mu$  handled analogously \*/
- 6:     Find  $r \in R_-$  with smallest  $p(L_r = 1 | \ell_{\setminus r}, \mathbf{x}; \mathbf{w}_z, \mathbf{w}_{\ell})$
- 7:     Set  $\ell_r \leftarrow +1$

- ▶ Does not seem to attempt a tradeoff between staying close to  $\mu$  and enhancing compatibility between  $\ell$  and  $y$
- ▶ I.e., **large** assumed to be *very* large
- ▶ Fixing  $\mathbf{z}$  and then  $\mathbf{w}_z, \mathbf{w}_{\ell}$  same as MIML-RE

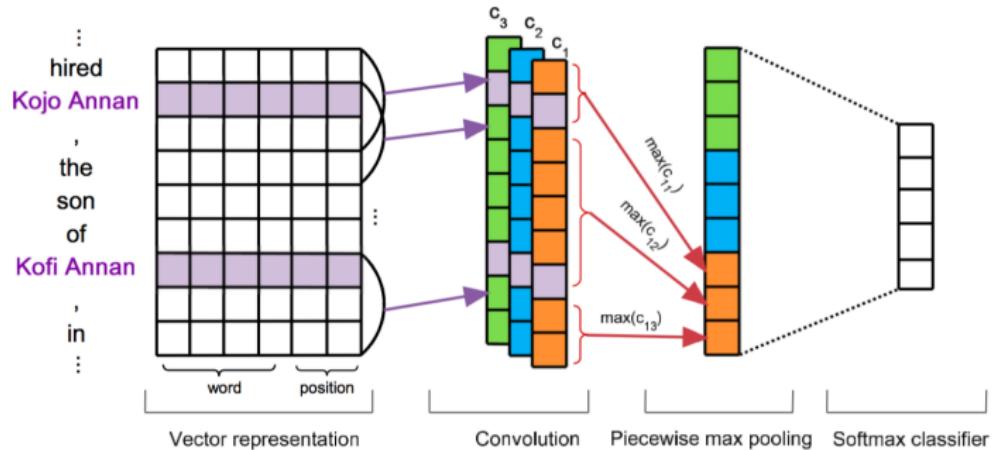
## Neural Networks for Distant Supervision

- ▶ Prior to Neural Networks, all techniques utilised carefully curated NLP features for relation prediction.
- ▶ Extraction of traditional NLP features may creep in additional errors into the pipeline.
- ▶ Sentences encountered in relation extraction problem are on an average more than 40 words, which might lead to higher errors in NLP feature extraction [9].
- ▶ To avoid errors in feature extraction from NLP pipeline, most modern systems use neural networks for extraction the features.

## Piecewise Convolution Neural Network (PCNN)

- ▶ PCNN adapts CNN-based relation extraction [10] to the distant supervision setting.
- ▶ CNNs are used to extract sentence features.
- ▶ Sentence features are then processed by a novel piecewise pooling method to preserve structural features of the sentence.
- ▶ Final features are then processed using a linear layer followed by softmax to generate probability of a given relation for an entity pair.

## Piecewise Convolution Neural Network (PCNN) (2)



**Figure:** The architecture of PCNNs used for distant supervised relation extraction, illustrating the procedure for handling one instance of a bag and predicting the relation between Kojo Annan and Kofi Annan (Image from original paper)

## Piecewise Convolution Neural Network (PCNN) (3)

- ▶ Let  $Q_{i:j} \in \mathbb{R}^{(j-i+1) \times d}$  be the concatenation of word vectors from  $q_i$  to  $q_j$
- ▶ Each word vector  $q_i$  is concatenation of word2vec embedding and the position embedding of the word.
- ▶ Convolution is a dot product between a filter  $\Omega \in \mathbb{R}^{n \times d}$  and each consecutive  $n$ -gram in a stacked sequence of word vectors  $Q$
- ▶ Dot product  $c_i \in \mathbb{R}$  is defined as:

$$c_i = \Omega \times Q_{i-n+1:i}, \quad 1 \leq i \leq k+n-1$$

- ▶ Resulting  $C_{\Omega,Q} = \langle c_1, \dots, c_{k+n-1} \rangle$  is the embedding for the sentence.
- ▶ Maxpooling helps NN to go from variable sized sentences to fixed sized representations.

## Piecewise Convolution Neural Network (PCNN) (4)

- ▶ To preserve structural features in a sentence, piecewise maxpooling is performed.
- ▶ Maxpooled  $p_{\Omega,Q} \in \mathbb{R}$  corresponding to filter  $w$  is obtained as follows.

$$p_{\Omega,Q} = \langle \max(s_{[o,e1]}), \max(s_{[e1,e2]}), \max(s_{[e2,-1]}) \rangle$$

where  $s_{[i,j]}$  is  $[c_i, c_{i+1} \dots c_j]$  and  $e1, e2$  are the locations of the entities in the sentence.

- ▶ Maxpooled sentence representation is further processed using non-linearity layer, linear layer, followed by softmax layer to predict the relation label. [9]

# PCNN Performance Comparison

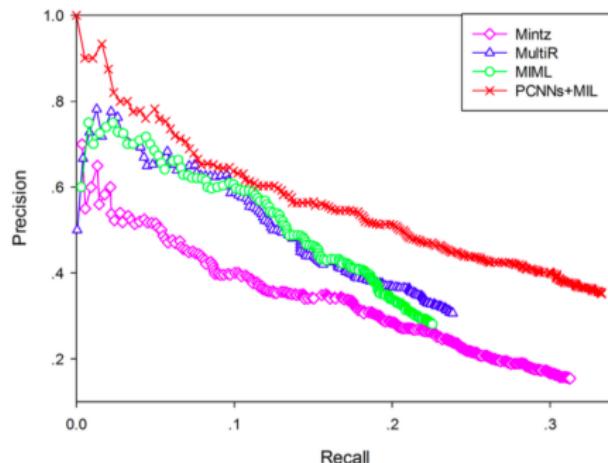


Figure: Performance comparison of the PCNN method with traditional approaches (Image from original paper)

## Neural Relation Extraction (NRE) with Selective Attention over Instances

- ▶ PCNN model used a single sentence with highest relation probability to predict the label for an instance-set.
- ▶ Using single sentence from a bag to predict relation label, misses crucial information from other sentences for multi-relation prediction.
- ▶ NRE [11] devises an attention mechanism to aggregate information from various sentence to form a single instance set representation.

## Neural Relation Extraction (NRE) with Selective Attention over Instances (2)

- ▶ Suppose instance set  $S$  contains  $n$  sentences  $x_1, x_2 \dots x_n$ .
- ▶ Representation for this instance set( $s$ ) is made as follows:

$$s = \sum \alpha_i x_i$$

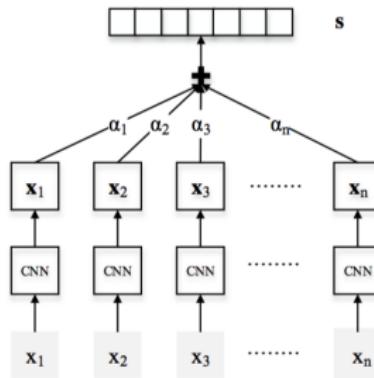
- ▶ Attention values  $\alpha$  is defined as:

$$\alpha_i = \frac{\exp(e_i)}{\sum_{1,k} \exp(e_k)}, e_i = x_i A r$$

Where,  $A, r$  are parameters to be learned.  $A$  is a weighted diagonal matrix, and  $r$  is the query vector associated with relation  $r$ .

- ▶ Similar to PCNN, instance set representation ( $s$ ) is then used as an input to a linear layer followed by softmax to predict multiple relations for each entity-pair.

# Neural Relation Extraction with Selective Attention over Instances



**Figure:** The architecture of sentence-level attention-based CNN, where  $x_i$  &  $x_i$  indicate the original sentence for an entity pair and its corresponding sentence representation,  $\alpha_i$  is the weight given by sentence-level attention, and  $s$  indicates the representation of the sentence set (Image from original paper)

## Neural Relation Extraction with Selective Attention over Instances

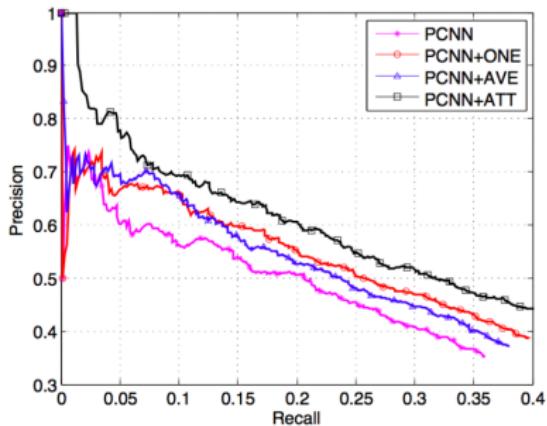


Figure: Performance comparison of the NRE method with traditional approaches (Image from original paper)

## References

- [1] R. C. Bunescu and R. J. Mooney, "A shortest path dependency kernel for relation extraction," in *EMNLP Conference*. ACL, 2005, pp. 724–731. [Online]. Available: <http://acl.ldc.upenn.edu/H/H05/H05-1091.pdf>
- [2] S. Brin, "Extracting patterns and relations from the World Wide Web," in *WebDB Workshop*, ser. LNCS, P. Atzeni, A. O. Mendelzon, and G. Mecca, Eds., vol. 1590. Valencia, Spain: Springer, Mar. 1998, pp. 172–183. [Online]. Available: <http://ilpubs.stanford.edu:8090/421/1/1999-65.pdf>
- [3] E. Agichtein and L. Gravano, "Snowball: Extracting relations from large plain-text collections," in *ICDL*, 2000, pp. 85–94. [Online]. Available: <http://www.academia.edu/download/31007490/cucs-033-99.pdf>
- [4] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *ACL Conference*, 2009, pp. 1003–1011. [Online]. Available: <http://www.aclweb.org/anthology/P09-1113>

## References (2)

- [5] M. Surdeanu, J. Tibshirani, R. Nallapati, and C. D. Manning, "Multi-instance multi-label learning for relation extraction," in *EMNLP Conference*, 2012, pp. 455–465. [Online]. Available: <http://anthology.aclweb.org/D/D12/D12-1042.pdf>
- [6] G. Angeli, J. Tibshirani, J. Wu, and C. D. Manning, "Combining distant and partial supervision for relation extraction." in *EMNLP Conference*, 2014, pp. 1556–1567. [Online]. Available: <http://www.anthology.aclweb.org/D/D14/D14-1164.pdf>
- [7] R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and D. S. Weld, "Knowledge-based weak supervision for information extraction of overlapping relations," in *ACL Conference*, 2011, pp. 541–550. [Online]. Available: <http://anthology.aclweb.org/P/P11/P11-1055.pdf>
- [8] B. Min, R. Grishman, L. Wan, C. Wang, and D. Gondek, "Distant supervision for relation extraction with an incomplete knowledge base." in *NAACL Conference*, 2013, pp. 777–782. [Online]. Available: <http://www.anthology.aclweb.org/N/N13/N13-1095.pdf>

## References (3)

- [9] D. Zeng, K. Liu, Y. Chen, and J. Zhao, "Distant supervision for relation extraction via piecewise convolutional neural networks," in *EMNLP Conference*, 2015, pp. 1753–1762. [Online]. Available: <http://www.aclweb.org/anthology/D15-1203>
- [10] D. Zeng, K. Liu, S. Lai, G. Zhou, J. Zhao *et al.*, "Relation classification via convolutional deep neural network." in *COLING*, 2014, pp. 2335–2344.
- [11] Y. Lin, S. Shen, Z. Liu, H. Luan, and M. Sun, "Neural relation extraction with selective attention over instances," in *ACL Conference*, August 2016, pp. 2124–2133. [Online]. Available: <http://www.aclweb.org/anthology/P16-1200>

# Open knowledge acquisition

(Includes slides from Partha Talukdar)

# Closed vs. open domain extraction

## Closed domain

- ▶ Finite closed set of entities and relations
- ▶ Entities and relations are normalized to IDs
- ▶ For each relation, labeled positive and negative tuples  $(e_s, r, e_o) \rightarrow \pm 1$  available
- ▶ Training effort scales with number of types

## Closed vs. open domain extraction (2)

### Open domain

- ▶ No predetermined type system for entities or relations
- ▶ Entities and relations remain textual, not canonicalized to IDs
- ▶ No labeled instances specific to entity and relation types
- ▶ Ideally want “constant” training effort as number of entity and relation types scale
- ▶ “Unsupervised semantic parsing”
- ▶ Less glamorously, clustering patterns between entity pairs (wrote a review, wrote a critique, reviewed) and using cluster IDs as relations
- ▶ More sophisticated: “Napoleon reviewed the Old Guard” vs. “Roger Ebert reviewed *The Fall*”

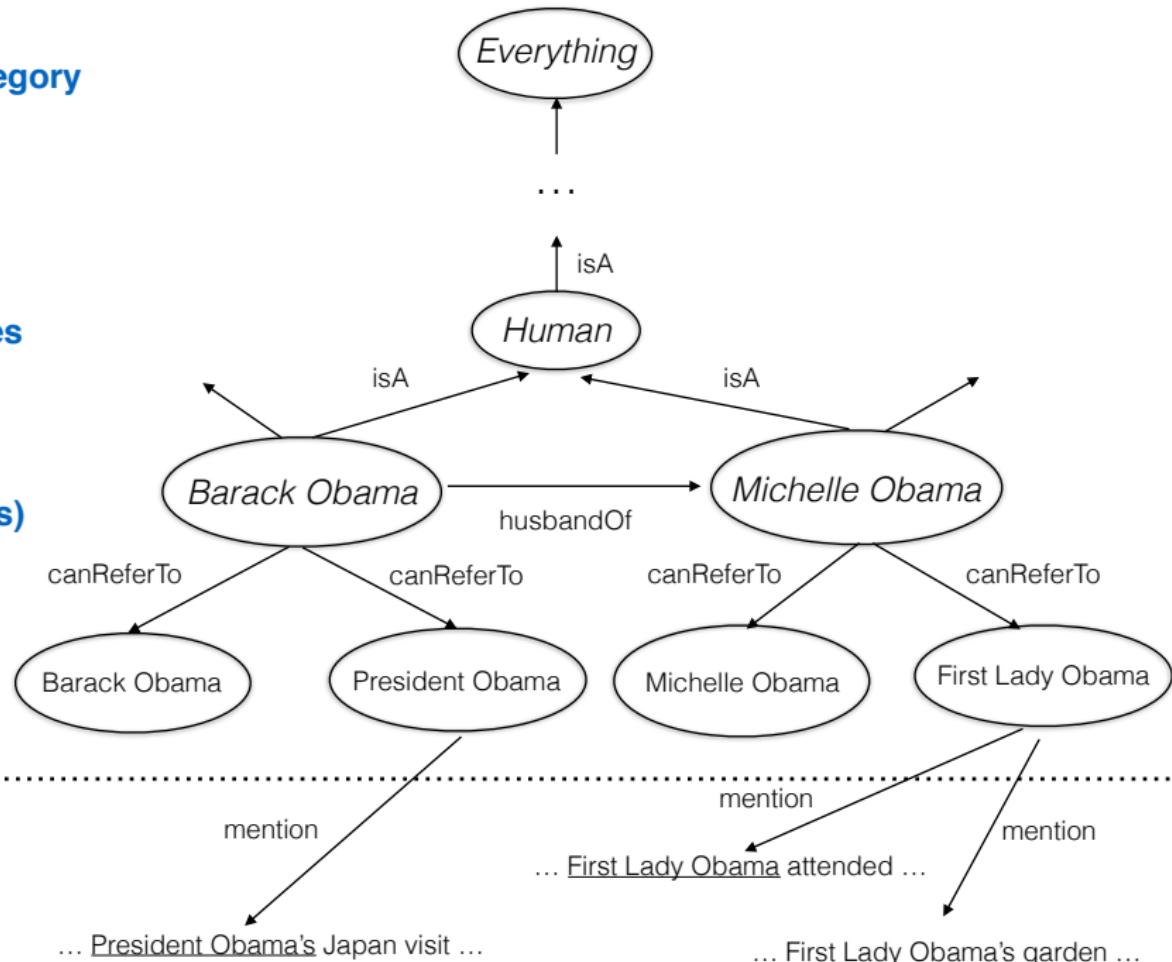
## Ontology Root Category

## Categories

## Entities (Concepts)

## Noun Phrases

## Text Corpus



# NELL: Never Ending Language Learner @ CMU

# NELL: Never Ending Language Learner @ CMU

Inputs:

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)
- few seed examples of each predicate

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)
- few seed examples of each predicate
- the web

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)
- few seed examples of each predicate
- the web
- occasional interaction with human trainers

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)
- few seed examples of each predicate
- the web
- occasional interaction with human trainers

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)
- few seed examples of each predicate
- the web
- occasional interaction with human trainers

The task:

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)
- few seed examples of each predicate
- the web
- occasional interaction with human trainers

The task:

- run 24x7, forever

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)
- few seed examples of each predicate
- the web
- occasional interaction with human trainers

The task:

- run 24x7, forever
- each day:

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)
- few seed examples of each predicate
- the web
- occasional interaction with human trainers

The task:

- run 24x7, forever
- each day:
  - extract more facts from the web

# NELL: Never Ending Language Learner @ CMU

Inputs:

- initial predicates (e.g., person, city, mayorOfCity, etc.)
- few seed examples of each predicate
- the web
- occasional interaction with human trainers

The task:

- run 24x7, forever
- each day:
  - extract more facts from the web
  - learn to read (perform #1) better than yesterday

# NELL Today

# NELL Today

Running 24x7, since January, 12, 2010

Result:

KB with > 100 million candidate beliefs, growing daily  
learning to reason, as well as read  
automatically extending its ontology

# NELL Today

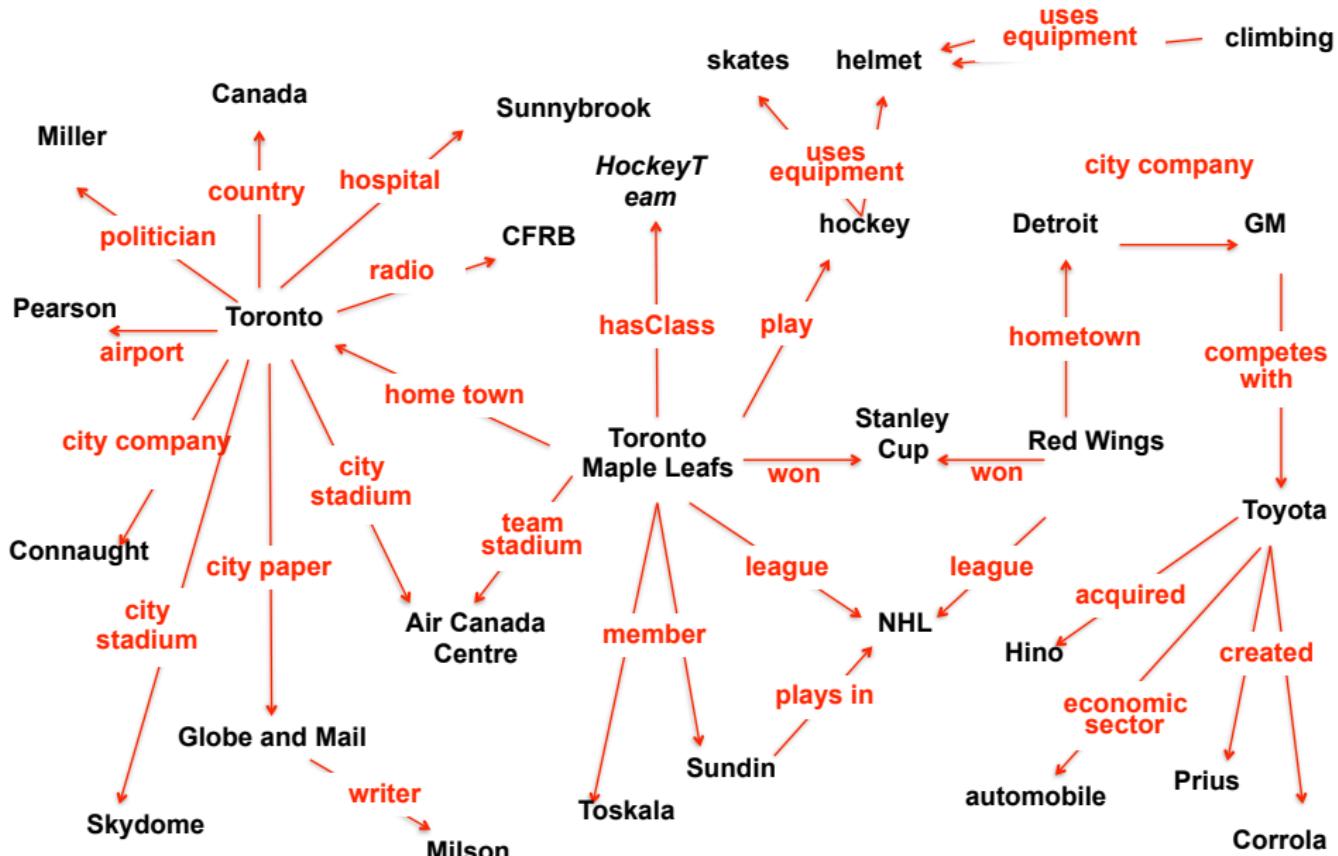
Running 24x7, since January, 12, 2010

Result:

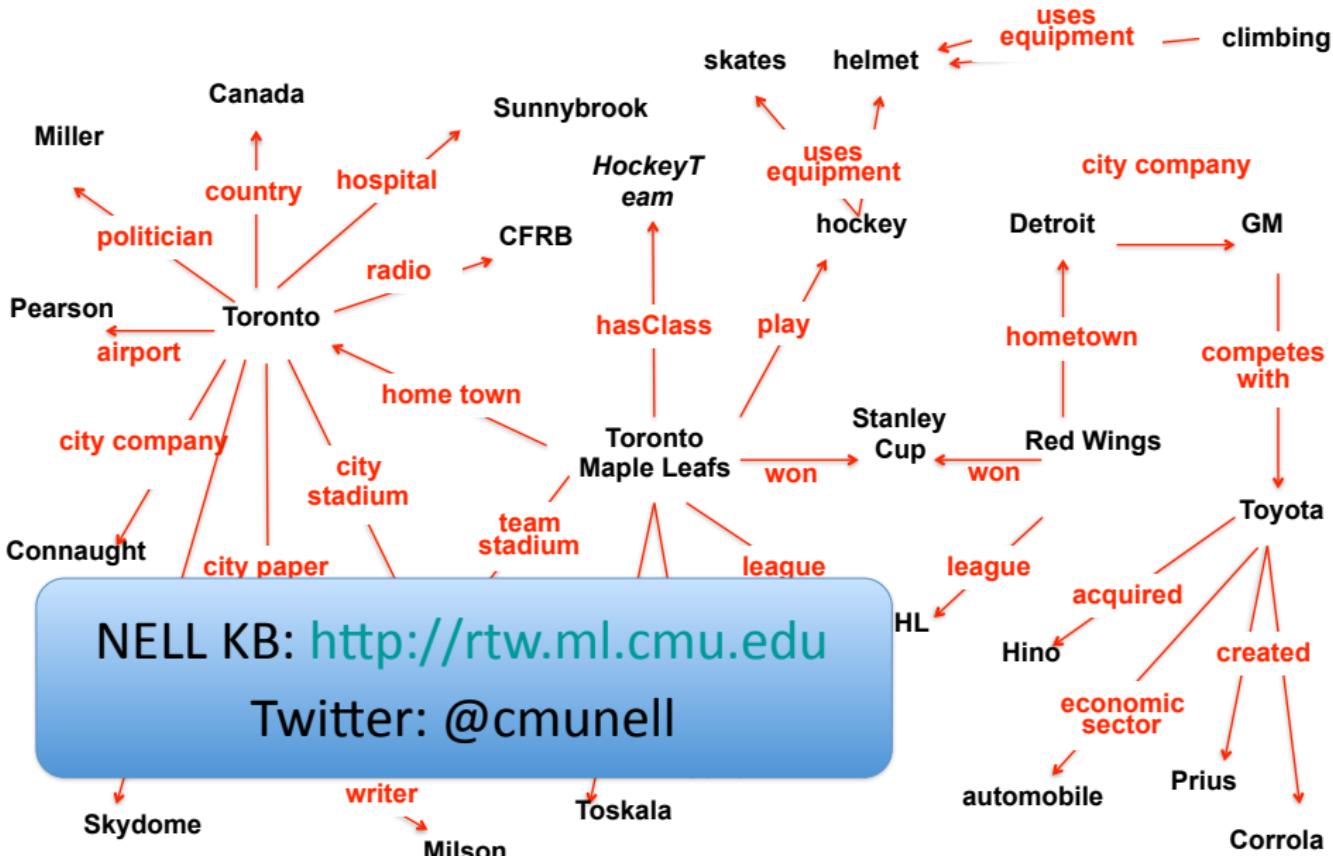
KB with > 100 million candidate beliefs, growing daily  
learning to reason, as well as read  
automatically extending its ontology



# NELL Knowledge Fragment



# NELL Knowledge Fragment



NELL KB: <http://rtw.ml.cmu.edu>

Twitter: @cmunell

# NELL Today

- eg. “[diabetes](#)”, “[Avandia](#)”, “[tea](#)”, “[IBM](#)”, “[love](#)” “[baseball](#)”  
“[BacteriaCausesCondition](#)” “[kitchenItem](#)” “[ClothingGoesWithClothing](#)” ...

## Recently-Learned Facts



instance	iteration	date learned
<a href="#">mark bellhorn</a> is a <a href="#">Mexican person</a>	763	27-aug-2013
<a href="#">methenamine mandelate tablet</a> is a <a href="#">drug</a>	763	27-aug-2013
<a href="#">pete zimmer</a> is a <a href="#">person</a>	763	27-aug-2013
<a href="#">sandhills clubtail</a> is a <a href="#">vertebrate</a>	764	31-aug-2013
<a href="#">jeffrey carlson</a> is a <a href="#">chef</a>	763	27-aug-2013
<a href="#">sutton</a> is a park <a href="#">in the city london</a>	767	06-sep-2013
<a href="#">pushkin</a> was born in <a href="#">moscow</a>	767	06-sep-2013
<a href="#">honda</a> is a company that <a href="#">produces accord</a>	766	04-sep-2013
<a href="#">spurs</a> is a sports team that <a href="#">plays against magic</a>	763	27-aug-2013
<a href="#">baseball</a> is a sport <a href="#">played in the venue ballpark in arlington</a>	766	04-sep-2013

# Other Related Efforts



High Supervision



**NELL**



Low Supervision

## Never-Ending Learning

T. Mitchell<sup>\*</sup>, W. Cohen<sup>\*</sup>, E. Hruschka<sup>\*†</sup>, P. Talukdar<sup>\*‡</sup>, J. Betteridge<sup>\*</sup>, A. Carlson<sup>\*§</sup>, B. Dalvi<sup>\*</sup>, M. Gardner<sup>\*</sup>, B. Kisiel<sup>\*</sup>, J. Krishnamurthy<sup>\*</sup>, N. Lao<sup>\*¶</sup>, K. Mazaitis<sup>\*</sup>, T. Mohamed<sup>\*</sup>, N. Nakashole<sup>\*</sup>, E. Platanios<sup>\*</sup>, A. Ritter<sup>\*¶</sup>, M. Samadi<sup>\*</sup>, B. Settles<sup>\*¶</sup>, R. Wang<sup>\*</sup>, D. Wijaya<sup>\*</sup>, A. Gupta<sup>\*</sup>, X. Chen<sup>\*</sup>, A. Saparov<sup>\*</sup>, M. Greaves<sup>\*||</sup>, J. Welling<sup>||</sup>

tom.mitchell@cs.cmu.edu

### Abstract

Whereas people learn many different types of knowledge from diverse experiences over many years, most current machine learning systems acquire just a single function or data model from just a single data set. We propose a *never-ending learning* paradigm for machine learning, to better reflect the more ambitious and encompassing type of learning performed by humans. As a case study, we describe the Never-Ending Language Learner (NELL) which achieves some of the desired properties of a never-ending learner, and we discuss lessons learned. NELL has been learning to read the web 24 hours/day since January 2010, and so far has acquired a knowledge base with over 80 million confidence-weighted beliefs (e.g., *servedWith(tea, biscuits)*). NELL has also learned millions of features and parameters that enable it to read these beliefs from the web. Additionally, it has learned to reason over these beliefs to infer new beliefs, and is able to extend its ontology by synthesizing new relational predicates. NELL can be tracked online at <http://rtw.ml.cmu.edu>, and followed on Twitter at @CMUNELL.

### Introduction

Machine learning is a highly successful branch of AI, and machine learning software is now widely used for tasks from spam filtering, to speech recognition, to credit card fraud detection, to face recognition. Despite this success, the ways in which computers learn today remain surprisingly narrow when compared to human learning. This paper explores an alternative paradigm for machine learning that more closely models the diversity, competence and cumulative nature of human learning. We call this alternative paradigm *never-ending learning*.

To illustrate, note that in each of the above examples the computer learns only a single function to perform a single

task in isolation, usually from human labeled training examples of inputs and outputs of that function. In spam filtering, for instance, training examples consist of specific emails and spam or not-spam labels for each. This style of learning is often called *supervised function approximation*, because the abstract learning problem is to approximate some unknown function  $f: X \rightarrow Y$  (e.g., the spam filter) given a training set of input/output pairs  $\{(x_i, y_i)\}$  of that function. Other machine learning paradigms exist as well (e.g., unsupervised clustering, topic modeling) but these paradigms also typically acquire only a single function or data model from a single dataset.

In contrast to these paradigms for learning single functions from well-organized data sets over short time-frames, humans learn many different functions (i.e., different types of knowledge) over years of accumulated diverse experience, using extensive background knowledge learned from earlier experiences to guide subsequent learning.

The thesis of this paper is that *we will never truly understand machine or human learning until we can build computer programs that, like people,*

- learn many different types of knowledge or functions,
- from years of diverse, mostly self-supervised experience,
- in a staged curricular fashion, where previously learned knowledge enables learning further types of knowledge,
- where self-reflection and the ability to formulate new representations and new learning tasks enable the learner to avoid stagnation and performance plateaus.

We refer to this learning paradigm as “never-ending learning.” The contributions of this paper are to (1) define more precisely the never-ending learning paradigm, (2) present as a case study a computer program called the Never-Ending Language Learner (NELL) which implements several of these capabilities, and which has been learning to read the web 24 hours/day for over four years, and (3) identify from NELL’s strengths and weaknesses a number of key design features important to any never-ending learning system.

### Related Work

Previous research has considered the problem of designing machine learning agents that persist over long periods of time (e.g., life long learning (Thrun and Mitchell 1995)), and that learn to learn (Thrun and Pratt 1998), yet there remain few if any working systems that demonstrate

<sup>\*</sup>Carnegie Mellon University, USA

<sup>†</sup>University of São Carlos, Brazil

<sup>‡</sup>Indian Institute of Science, India

<sup>§</sup>Google Inc., USA

<sup>¶</sup>Research carried out while at Carnegie Mellon University

<sup>||</sup>Ohio State University, USA

<sup>††</sup>Dublin City University, Ireland

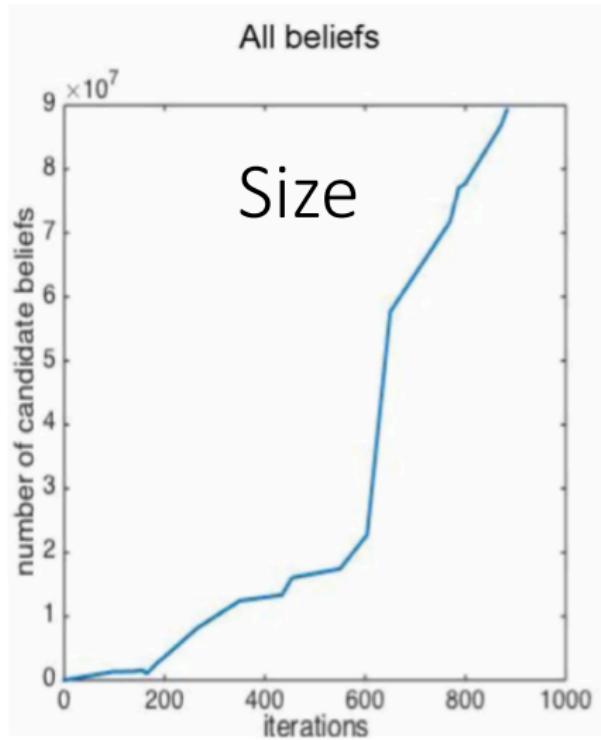
<sup>†††</sup>Alpine Data Labs, USA

<sup>††††</sup>Pittsburgh Supercomputing Center, USA

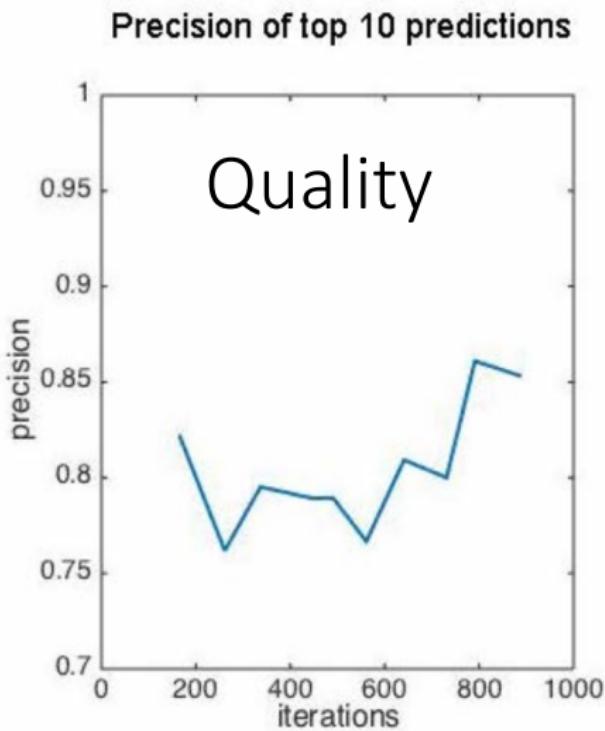
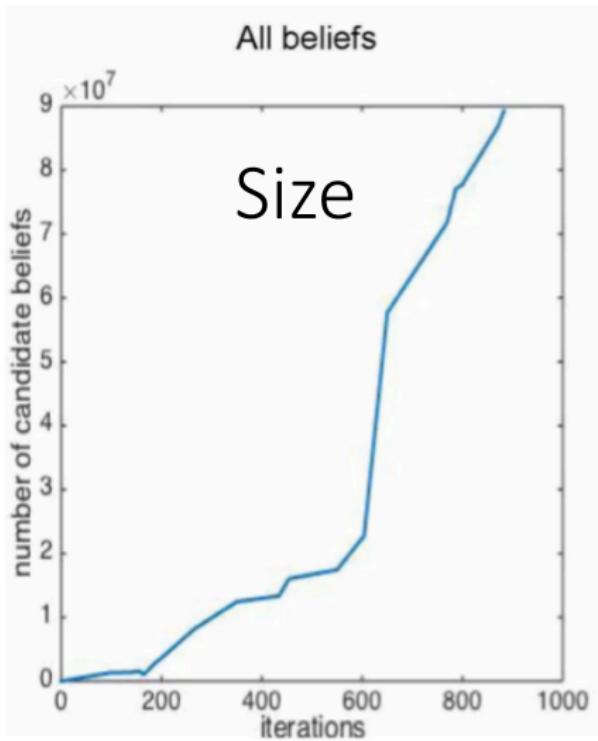
Copyright © 2013, Association for the Advancement of Artificial Intelligence ([www.aaai.org](http://www.aaai.org)). All rights reserved.

# NELL's Growth over Time

# NELL's Growth over Time



# NELL's Growth over Time



# How does NELL work?

# Semi-Supervised Bootstrap Learning

Learn which  
noun phrases  
are cities:

Paris  
Pittsburgh  
Seattle  
Montpelier

# Semi-Supervised Bootstrap Learning

Learn which  
noun phrases  
are cities:

Paris  
Pittsburgh  
Seattle  
Montpelier



mayor of arg1  
live in arg1

# Semi-Supervised Bootstrap Learning

Learn which  
noun phrases  
are cities:

Paris  
Pittsburgh  
Seattle  
Montpelier

San Francisco  
Berlin  
denial



mayor of arg1  
live in arg1

# Semi-Supervised Bootstrap Learning

Learn which  
noun phrases  
are cities:

Paris  
Pittsburgh  
Seattle  
Montpelier

San Francisco  
Berlin  
denial



mayor of arg1  
live in arg1



arg1 is home of  
traits such as arg1

# Semi-Supervised Bootstrap Learning

Learn which  
noun phrases  
are cities:

Paris  
Pittsburgh  
Seattle  
Montpelier

San Francisco  
Berlin  
denial

anxiety  
selfishness  
London



mayor of arg1  
live in arg1

arg1 is home of  
traits such as arg1

# Semi-Supervised Bootstrap Learning

Learn which  
noun phrases  
are cities:

Paris  
Pittsburgh  
Seattle  
Montpelier

San Francisco  
Berlin  
denial

anxiety  
selfishness  
London

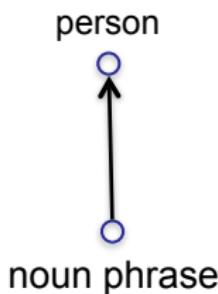


mayor of arg1  
live in arg1

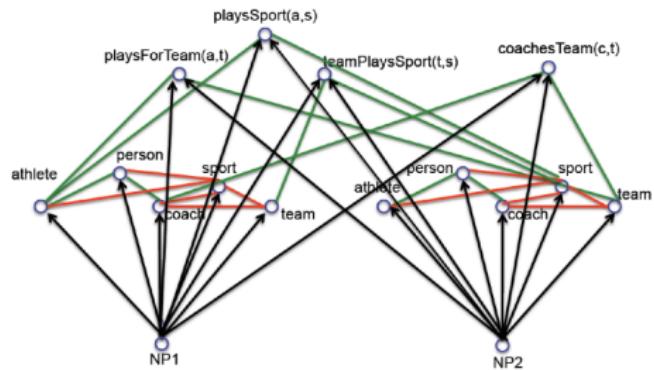


it's underconstrained!!

# Key Idea 1: Coupled semi-supervised training of many functions



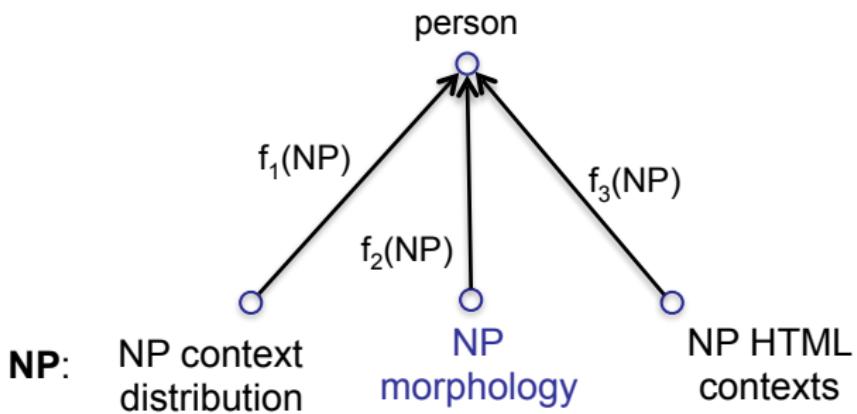
**hard**  
(underconstrained)  
semi-supervised  
learning problem



**much easier** (more constrained)  
semi-supervised learning problem

# Type 1 Coupling: Co-Training, Multi-View Learning

[Blum & Mitchell; 98]  
[Dasgupta et al; 01 ]  
[Ganchev et al., 08]  
[Sridharan & Kakade, 08]  
[Wang & Zhou, ICML10]



\_ is a friend  
rang the \_

capitalized?  
ends with ‘...ski’?

...  
\_ walked in

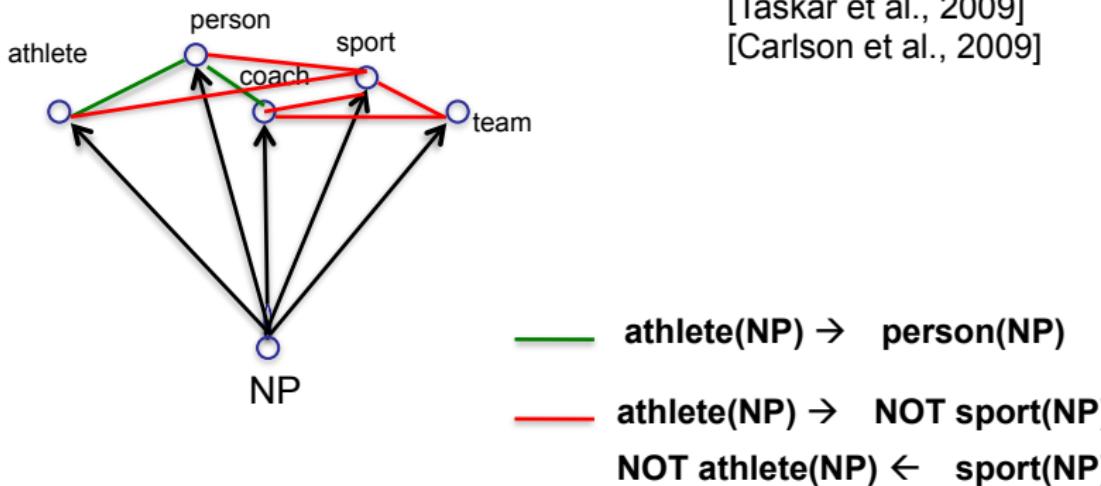
...  
contains “univ.”?

[www.celebrities.com](http://www.celebrities.com):  
<li> \_ </li>

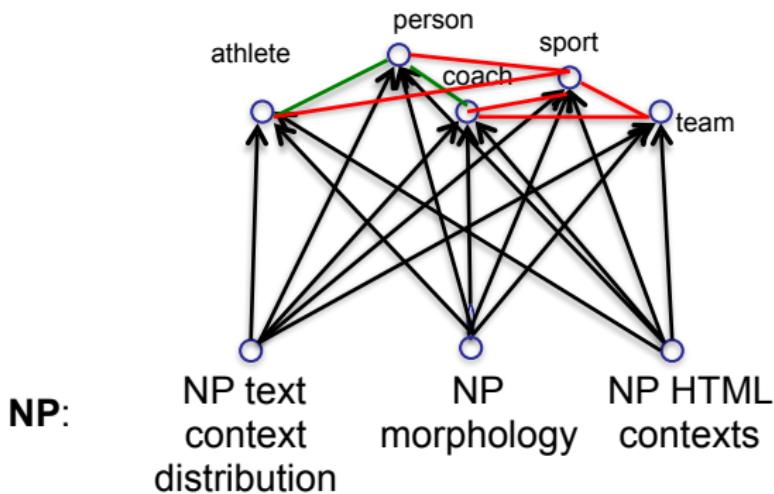
...

# Type 2 Coupling: Multi-task, Structured Outputs

[Daume, 2008]  
[Bakhir et al., eds. 2007]  
[Roth et al., 2008]  
[Taskar et al., 2009]  
[Carlson et al., 2009]

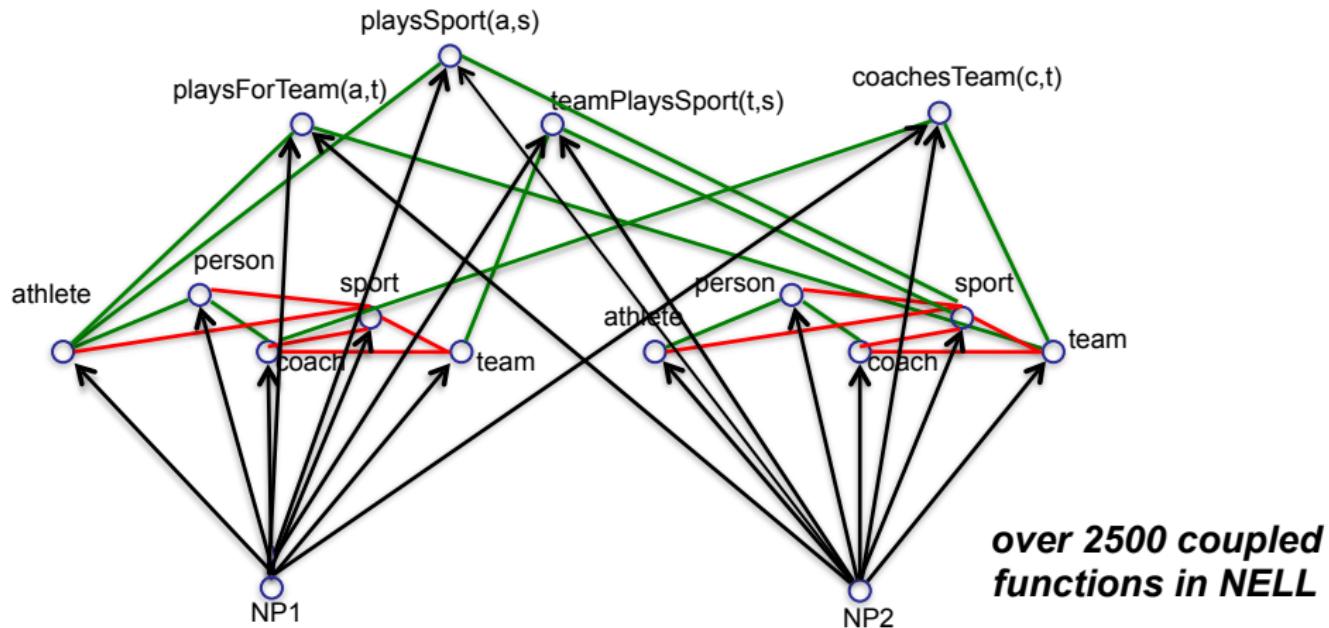


# Multi-view, Multi-Task Coupling

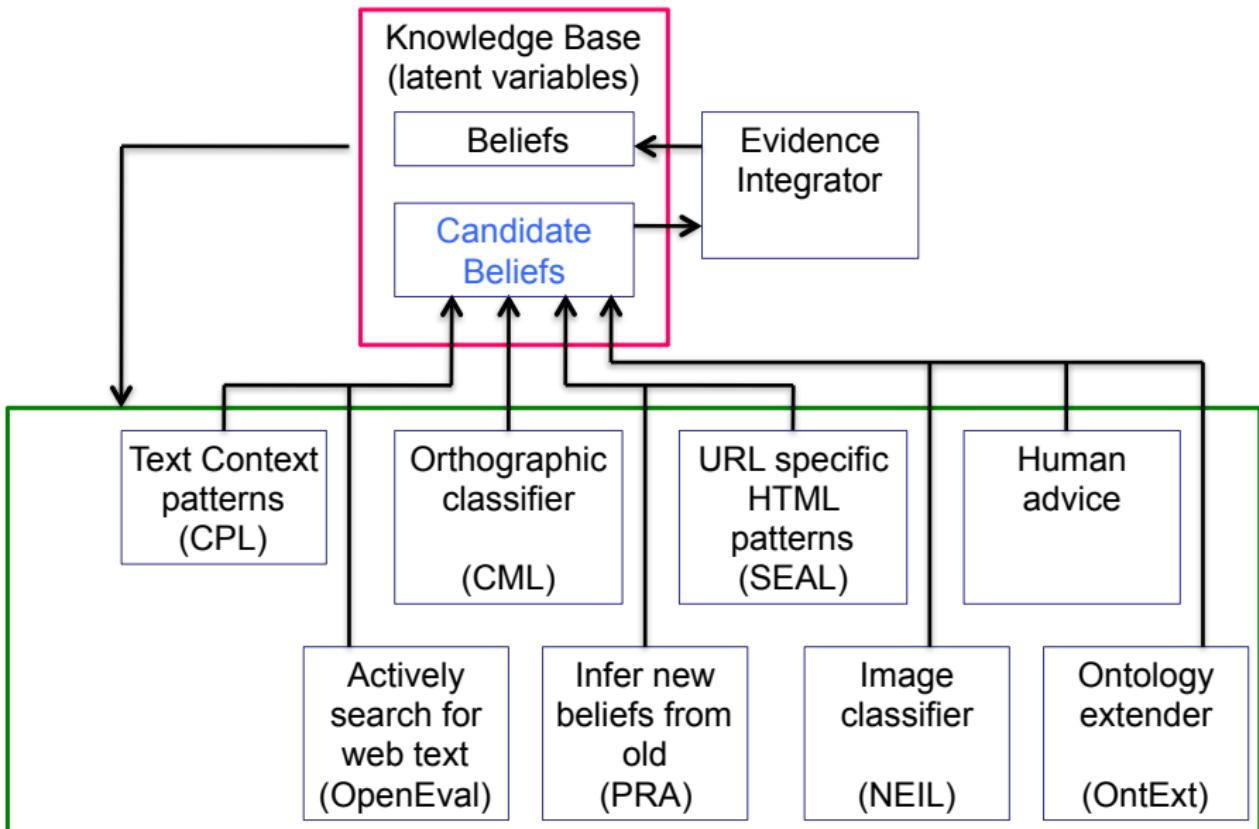


# Type 3 Coupling: Argument Types

$\text{playsSport}(\text{NP1}, \text{NP2}) \rightarrow \text{athlete}(\text{NP1}), \text{sport}(\text{NP2})$



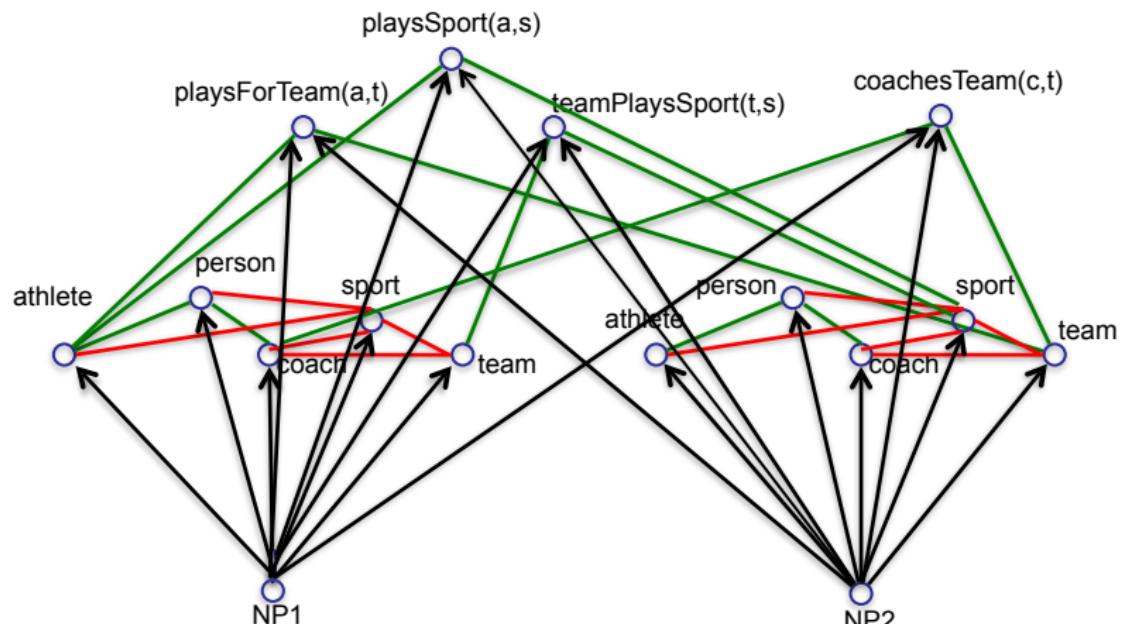
# NELL Architecture



If coupled learning is the key,  
how can we get new coupling constraints?

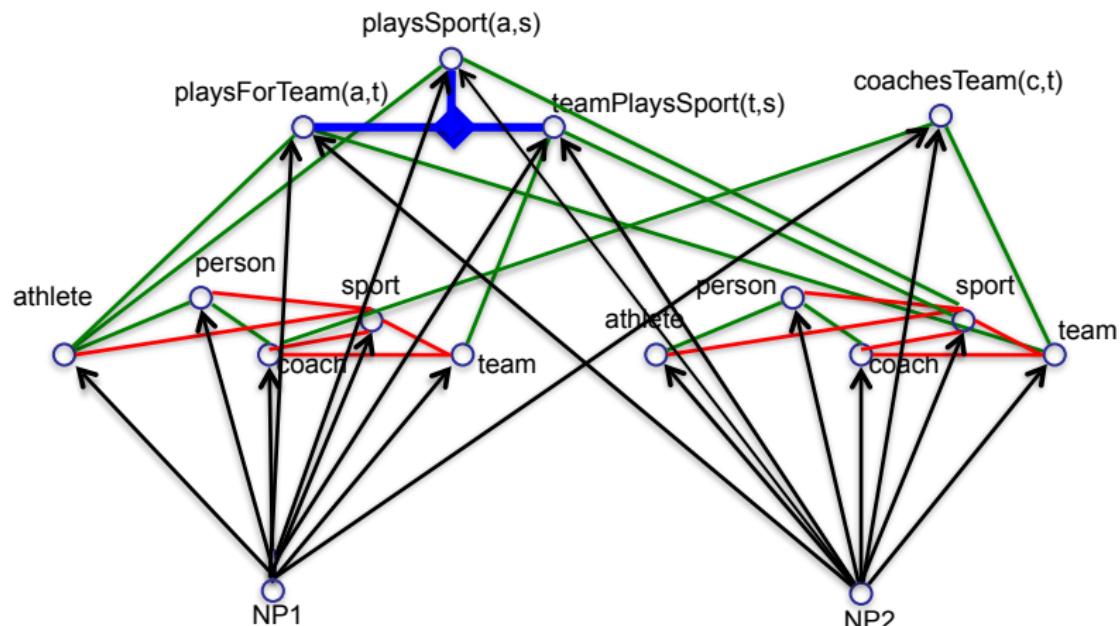
# Learned Probabilistic Horn Clause Rules

0.93  $\text{playsSport}(\text{x}, \text{y}) \leftarrow \text{playsForTeam}(\text{x}, \text{z}), \text{teamPlaysSport}(\text{z}, \text{y})$



# Learned Probabilistic Horn Clause Rules

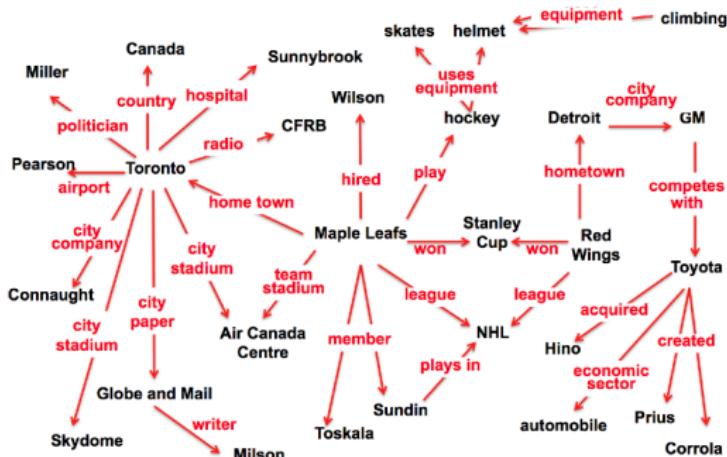
0.93  $\text{playsSport}(\text{x}, \text{y}) \leftarrow \text{playsForTeam}(\text{x}, \text{z}), \text{teamPlaysSport}(\text{z}, \text{y})$



# Inference by KB Random Walks

[Lao et al, EMNLP 2011]

KB:

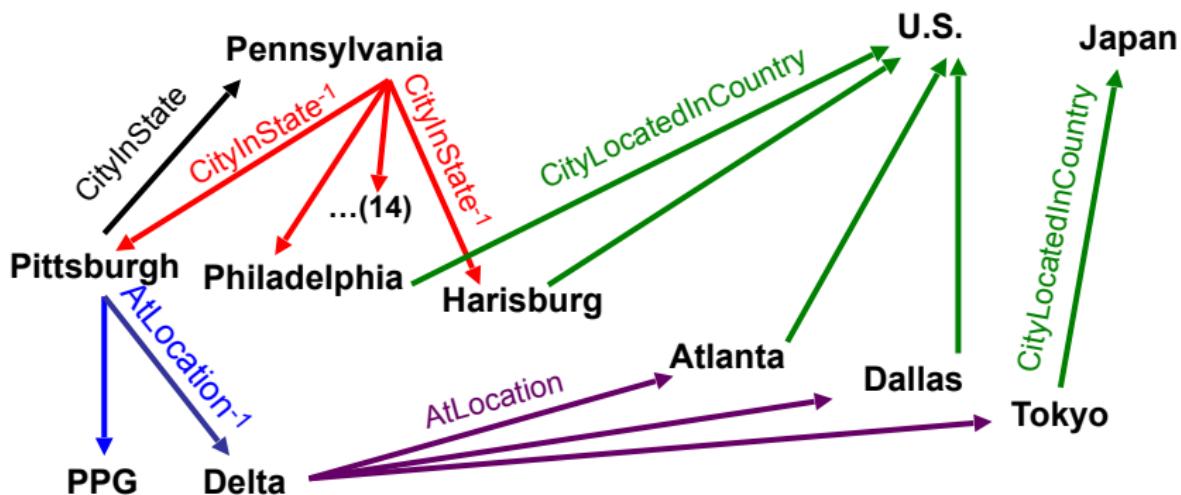


Random walk  
path type:

x — competes with → ? — economic sector → y

model  $\Pr(R(x,y))$ : logistic function for  $R(x,y)$

$i^{th}$  feature: probability of arriving at node y starting at node x, and taking a random walk along path type i

Feature = Typed PathCityInState, CityInState<sup>-1</sup>, CityLocatedInCountryAtLocation<sup>-1</sup>, AtLocation, CityLocatedInCountry

...

Feature Value

0.8

0.6

...

Logistic Regression Weight

0.32

0.20

...

CityLocatedInCountry(Pittsburgh) = U.S. p=0.58

# Random walk inference: learned path types

CityLocatedInCountry(*city, country*):

8.04 cityliesonriver, cityliesonriver<sup>-1</sup>, citylocatedincountry

5.42 hasofficeincity<sup>-1</sup>, hasofficeincity, citylocatedincountry

4.98 cityalsoknownas, cityalsoknownas, citylocatedincountry

2.85 citycapitalofcountry, citylocatedincountry<sup>-1</sup>, citylocatedincountry

2.29 agentactsinlocation<sup>-1</sup>, agentactsinlocation, citylocatedincountry

1.22 statehascapital<sup>-1</sup>, statelocatedincountry

0.66 citycapitalofcountry

7 of the 2985 learned paths for CityLocatedInCountry

Key Idea 3:  
Automatically extend ontology

# Example Discovered Relations

[Mohamed et al. EMNLP 2011]

Category Pair	Frequent Instance Pairs	Text Contexts	Suggested Name
MusicInstrument Musician	sitar, George Harrison tenor sax, Stan Getz trombone, Tommy Dorsey vibes, Lionel Hampton	ARG1 master ARG2 ARG1 virtuoso ARG2 ARG1 legend ARG2 ARG2 plays ARG1	Master
Disease Disease	pinched nerve, herniated disk tennis elbow, tendonitis blepharospasm, dystonia	ARG1 is due to ARG2 ARG1 is caused by ARG2	IsDueTo
CellType Chemical	epithelial cells, surfactant neurons, serotonin mast cells, histamine	ARG1 that release ARG2 ARG2 releasing ARG1	ThatRelease
Mammals Plant	koala bears, eucalyptus sheep, grasses goats, saplings	ARG1 eat ARG2 ARG2 eating ARG1	Eat
River City	Seine, Paris Nile, Cairo Tiber river, Rome	ARG1 in heart of ARG2 ARG1 which flows through ARG2	InHeartOf

# Key Idea 4: Cumulative, Staged Learning

Learning X improves ability to learn Y

1. Classify noun phrases (NP's) by category
2. Classify NP pairs by relation
3. Discover rules to predict new relation instances
4. Learn which NP's (co)refer to which latent concepts
5. Discover new relations to extend ontology
6. Learn to infer relation instances via targeted random walks
7. Learn to assign temporal scope to beliefs
8. Learn to microread single sentences

---

9. Vision: co-train text and visual object recognition
10. Goal-driven reading: predict, then read to corroborate/correct
11. Make NELL a conversational agent on Twitter
12. Add a robot body to NELL

# NELL Summary

- Learning
  - Coupled multi-task, multi-view semi-supervised training
- Inference
  - Data mine the KB to learn inference rules
  - Scalable any-time inference via random walks
- Representation
  - Ontology extension
    - invent new categories and relations
    - combine statistical clustering with direct reading
  - Infer millions of latent concepts from observable text
- Curriculum
  - learn easiest things first, build on those to “learn to learn”

# Machine Reading at Web Scale

- A “universal schema” is impossible

# Machine Reading at Web Scale

- A “universal schema” is impossible
- Global consistency is like world peace

# Machine Reading at Web Scale

- A “universal schema” is impossible
- Global consistency is like world peace
- Ontological “glass ceiling”
  - Limited vocabulary
  - Pre-determined predicates
  - Swamped by reading at scale!



# Open IE Guiding Principles

- Domain independence
  - Training for each domain/fact type not feasible
- Scalability
  - Ability to process large number of documents fast
- Coherence
  - Readability important for human interactions

## Open information extraction<sup>8</sup>

- ▶ Traditional heavily-supervised IE is narrow
- ▶ Applied to small, homogeneous corpora
- ▶ On the Web
  - ▶ No parser is sufficiently accurate
  - ▶ No pre-trained named-entity taggers
  - ▶ Supervised learning is impractical
- ▶ Semisupervised learning needs a few hand-labeled examples **per concept**
- ▶ Concepts themselves are pre-specified

---

<sup>8</sup>From Etzioni slides at WSDM 2008

## TextRunner<sup>9</sup>

- ▶ Use a simple model of relationships in English to label extractions
- ▶ Bootstrap a general model of relationships in English sentences, encoded as a CRF
- ▶ Using a shallow parser, decompose each sentence into one or more (NP1, VP, NP2) chunks
- ▶ Use CRF model to retain relevant parts of each NP and VP

---

<sup>9</sup>From Etzioni slides at WSDM 2008

## TextRunner example<sup>10</sup>

- ▶ Extract Triple representing binary relation (**Arg1**, **Relation**, **Arg2**) from sentence.
- ▶ Internet powerhouse, EBay, was originally founded by Pierre Omidyar.
- ▶ Internet powerhouse, **EBay**, was originally **founded by** **Pierre Omidyar**.
- ▶ RDF-like triple (**Ebay**, **Founded by**, **Pierre Omidyar**)

---

<sup>10</sup>From Etzioni slides at WSDM 2008

## TextRunner example<sup>10</sup>

- ▶ Extract Triple representing binary relation (**Arg1**, **Relation**, **Arg2**) from sentence.
- ▶ Internet powerhouse, EBay, was originally founded by Pierre Omidyar.
- ▶ Internet powerhouse, **EBay**, was originally **founded by** **Pierre Omidyar**.
- ▶ RDF-like triple (**Ebay**, **Founded by**, **Pierre Omidyar**)

---

<sup>10</sup>From Etzioni slides at WSDM 2008

## TextRunner example<sup>10</sup>

- ▶ Extract Triple representing binary relation (**Arg1**, **Relation**, **Arg2**) from sentence.
- ▶ Internet powerhouse, EBay, was originally founded by Pierre Omidyar.
- ▶ Internet powerhouse, **EBay**, was originally **founded by** **Pierre Omidyar**.
- ▶ RDF-like triple (**Ebay**, **Founded by**, **Pierre Omidyar**)

---

<sup>10</sup>From Etzioni slides at WSDM 2008

## TextRunner example<sup>10</sup>

- ▶ Extract Triple representing binary relation (**Arg1**, **Relation**, **Arg2**) from sentence.
- ▶ Internet powerhouse, EBay, was originally founded by Pierre Omidyar.
- ▶ Internet powerhouse, **EBay**, was originally **founded by** **Pierre Omidyar**.
- ▶ RDF-like triple (**Ebay**, **Founded by**, **Pierre Omidyar**)

---

<sup>10</sup>From Etzioni slides at WSDM 2008

# Open Information Extraction

Extracting information from natural language text  
for *all* relations in *all* domains in a *few* passes.

*"When Saddam Hussain invaded Kuwait in 1990, the international.."*

↓ Open IE

(Saddam Hussain, invaded, Kuwait)

(Google, acquired, Youtube)  
(Oranges, contain, Vitamin C)  
(Edison, invented, phonograph)

...

Argument 1:  Relation:  Argument 2:

antibiotics (381)

Chlorine (113)

Ozone (61)

Heat (60)

Honey (55)

Benzoyl peroxide (45)

The heat kills the bacteria .

Heat kills the bacteria .

The heat kills bacteria .

Only heat kills bacteria .

Heat kills most bacteria .

Heat can kill the bacteria .

Heat will kill bacteria .

The high heat will kill bacteria .

Heat does kill bacteria .

# Open vs. Traditional IE

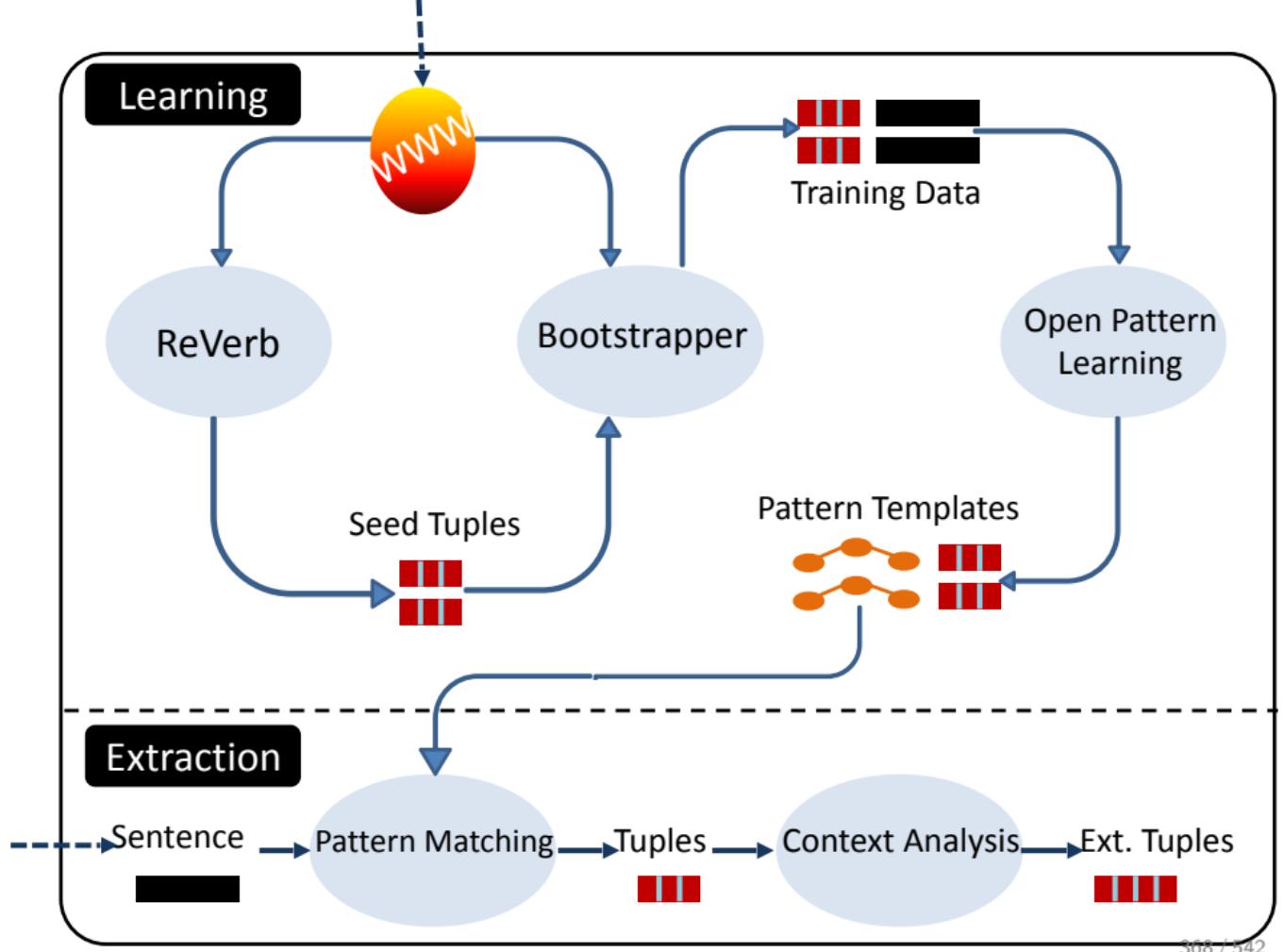
	Traditional IE	Open IE
<b>Input:</b>	Corpus + Hand-labeled Data	Corpus + Existing resources
<b>Relations:</b>	Specified in Advance	Discovered Automatically
<b>Complexity:</b>	$O(D * R)$ <i>R relations</i>	$O(D)$ <i>D documents</i>
<b>Output:</b>	relation-specific	Relation-independent

# Open Information Extraction

- 2007: Texrunner (~Open IE 1.0)
  - CRF and self-training
- 2010: ReVerb (~Open IE 2.0)
  - POS-based relation pattern
- 2012: OLLIE (~Open IE 3.0)
  - Dep-parse based extraction; nouns; attribution
- 2014: Open IE 4.0
  - SRL-based extraction; temporal, spatial...
- 2016 [@IITD]: Open IE 5.0
  - compound noun phrases, numbers, lists



increasing  
precision,  
recall,  
expressiveness



# Context Analysis

*"John refused to visit Vegas."*



(John, refused to visit, Vegas)

*"Early astronomers believed that the earth is the center of the universe."*



[(earth, is the center of, universe) Attribution: early astronomers]

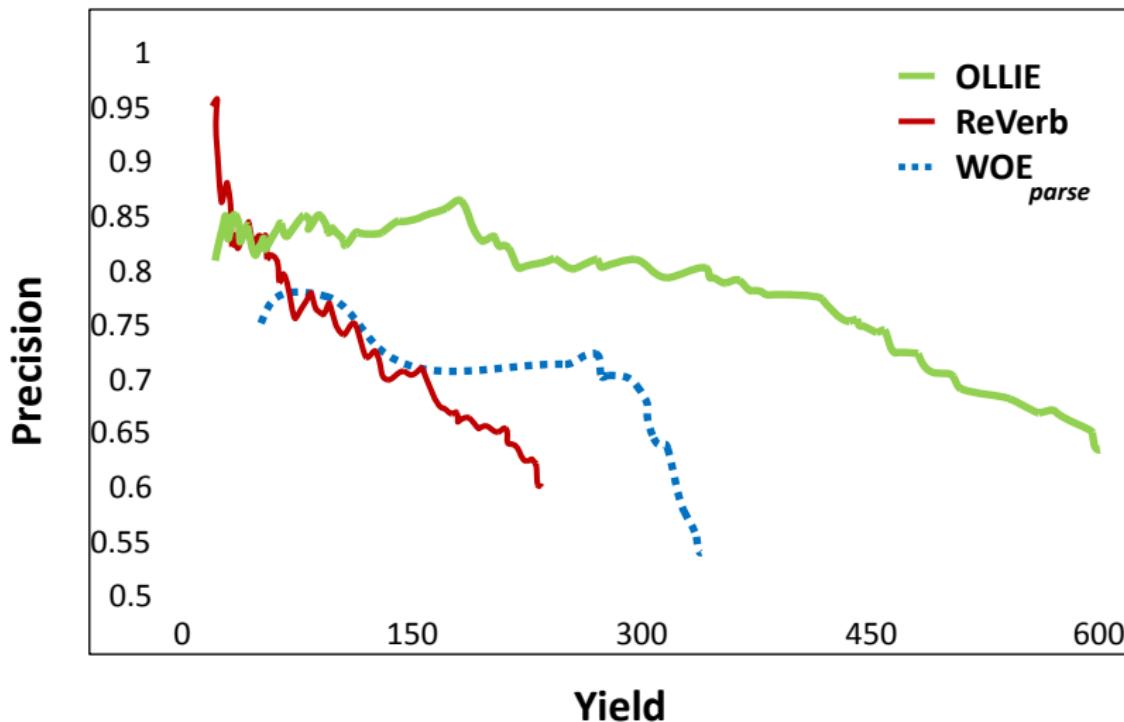
*"If she wins California, Hillary will be the nominated presidential candidate."*



[(Hillary, will be nominated, presidential candidate) Modifier: if she wins California]

# Evaluation

[Mausam, Schmitz, Bart, Soderland, Etzioni - EMNLP'12]



# Numerical Open IE

[Saha, Pal, Mausam ACL'17]

*"Venezuela with its inflation rate 96% is suffering from a major..."*



Numerical Open IE

(Venezuela, inflation rate, 96 %)

*"Grand Trunk Road is 1,005 kms long."*



Numerical Open IE

(Grand Trunk Road, has length, 1005 kms)

**OpenIE v5:**

<https://github.com/dair-iitd/OpenIE-standalone>

## TextRunner/OpenIE discussion

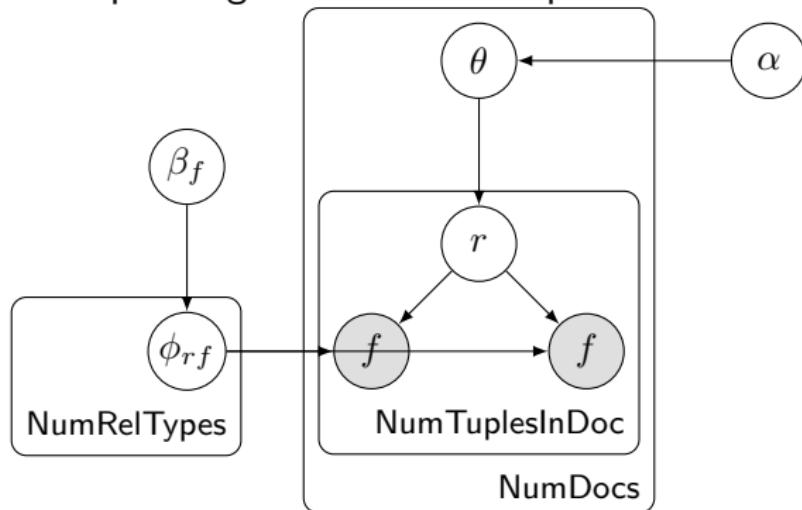
- ▶ Think of OpenIE triples as
  - ▶ Augmentations to KG inferred using redundancy/compressibility of KG, and
  - ▶ Aggregated facts from semi-digested corpus
- The latter view may be safer for downstream search tools
- ▶ No “correct” or “best” number of clusters for textual expression of types or relations
- ▶ May be more useful if entities are canonicalized into IDs
- ▶ Coref may help a little bit, too

## Generative OpenIE model: Rel-LDA

- ▶ For each document, use  $\text{Dir}(\alpha)$  to draw a  $\text{Multi}(\theta)$  distribution over relations (“topics”)
- ▶ To generate a relation tuple (usually, triple), first draw  $r$  using  $\text{Multi}(\theta)$

## Generative OpenIE model: Rel-LDA (2)

- ▶ For each triple, we have to generate (observed) features corresponding to the textual expression of  $r$



- ▶  $\{f\}$  includes observed mention features of subject and object entities, and features observed in the mention of  $r$

## Generative OpenIE model: Rel-LDA (3)

- During model fitting,  $\{f\}$  are seen;  $\alpha, \beta_f, \phi_{rf}$  are globally estimated;  $\theta, r$  are estimated for each document and tuple

E-step:

$$\Pr(r|\{f\}) \propto \Pr(r) \prod_f \Pr(f|r)$$
$$\propto (\alpha_r + n_{r|d}) \prod_f \frac{\beta_f + n_{f|r}}{\sum_{f'} (\beta_{f'} + n_{f'|r})}$$

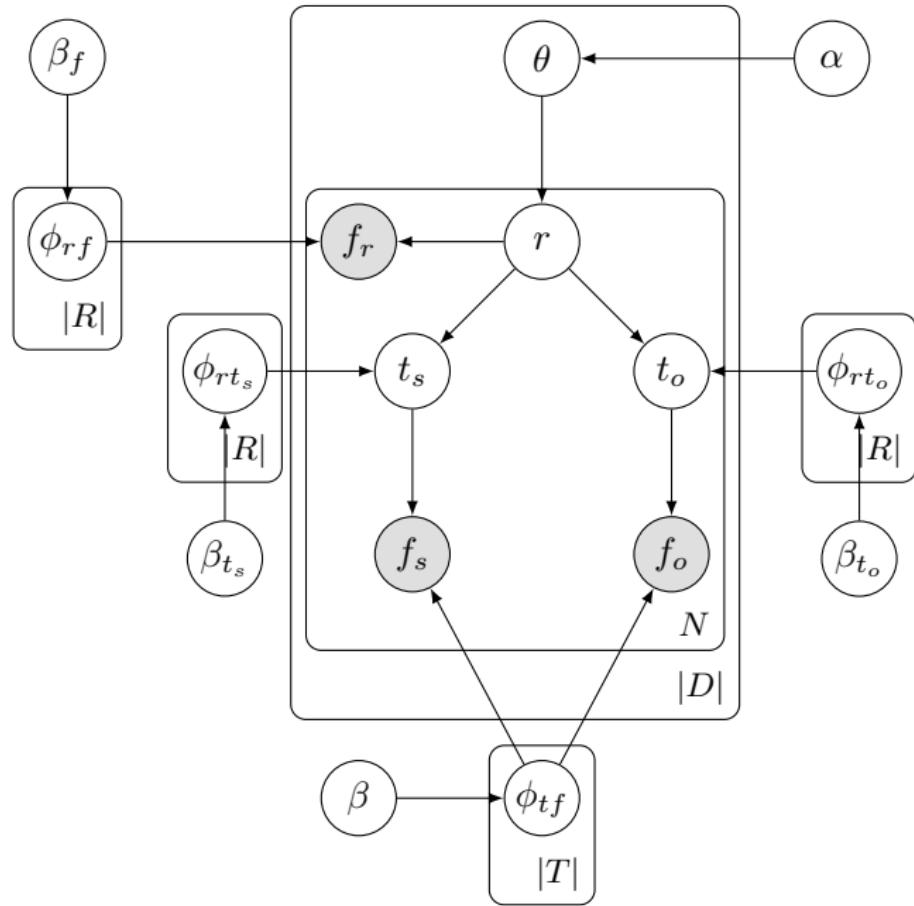
M-step:

$$\theta_{\text{doc}} = \frac{\alpha + n_{r|\text{doc}}}{\sum_{r'} (\alpha + n_{r'|\text{doc}})}$$
$$\phi_{rf} = \frac{\beta_f + n_{f|r}}{\sum_{f'} (\beta_{f'} + n_{f'|r})}$$

## Type-LDA

- ▶ Relations can hold only between certain entity types (selectional preference)
- ▶ Features of a tuple are grouped into relation-level and entity-level features
- ▶ Three different generators for these three groups of features
- ▶ Note that  $r$  and  $t$  are opaque indices (like cluster IDs), not directly equivalent to any named, understood relation or entity types
- ▶ Estimating correlations between  $r$  and  $f_r$ ,  $r$  and  $t$ ,  $t$  and  $f_s$ ,  $f_o$  can help “identify” some clusters in readable terms

## Type-LDA (2)



## Universal schema

- ▶ Learn to live with a combination of messy text and clean(er) structured KG info
- ▶ Implications are mostly probabilistic
- ▶ Do not try to predict semantic implication or equivalence deterministically
- ▶ Instead predict source data, both clean and messy, probabilistically
- ▶ Associate vector  $e_{s,o} \in \mathbb{R}^D$  with entity pair  $(e_s, e_o)$
- ▶ After absorbing structured and unstructured inputs to a common denominator, relations, entities, entity pairs, types are all free-text strings
- ▶ I.e., we will associate separate vectors to (Einstein, relativity), (Albert Einstein, theory of relativity) and (/m/0jcx, /m/07cwn)
- ▶ Expect these vectors to be drawn close via optimization

## Universal schema (2)

- ▶ Start with the **F model** (matrix factorization):

$$\Pr(e_s, r, e_o) = \sigma(\theta_{e_s, r, e_o}^F) = \sigma(\mathbf{r} \cdot \mathbf{e}_{s,o})$$

where  $\sigma(\bullet) = 1/(1 + e^{-\bullet})$  is the sigmoid function

- ▶ **Neighborhood model**: interpolate confidence for  $(e_s, r, e_o)$  based on confidence in other similar relations  $r'$  for  $(e_s, r', e_o)$

$$\theta_{e_s, r, e_o}^N = \sum_{(e_s, r', e_o) \notin \mathcal{O}} w_{r, r'}$$

where  $\mathcal{O}$  is the set of observed  $(e_s, r, e_o)$  and  $w_{r, r'}$  is a “directed association strength” between  $r$  and  $r'$

- ▶ Not clear how  $\mathbf{r}$  and  $\mathbf{e}_{s,o}$  contribute to  $w_{r, r'}$
- ▶ Anyway,  $w_{r, r'}$  is never mentioned in the **paper** again
- ▶ **Type model**: For each entity  $e$ , introduce latent vector  $\mathbf{t}_e \in \mathbb{R}^K$

## Universal schema (3)

- ▶ Again, the types of Einstein, Albert Einstein and /m/0jcx are given separate vectors; they might drift together through optimization
- ▶ Relation  $r$  takes  $\text{arity}(r)$  arguments
- ▶ So far we have modeled  $\text{arity}(r) = 2$  ( $s$  and  $o$ )
- ▶ For each relation  $r$  and argument slot  $i$ , introduce a vector  $\mathbf{d}_{r,i} \in \mathbb{R}^K$
- ▶ Recall even  $r$  is free-text; discovered, developed and invented are given separate vectors for each argument
- ▶ For specific tuple  $(r; e_1, e_2)$ , define

$$\theta_{r;e_1,e_2}^E = \sum_{i=1}^{\text{arity}(r)} \mathbf{d}_{r,i} \cdot \mathbf{t}_{e_i}$$

- ▶ Effectively a back-off protocol from entity vectors to type vectors

## Universal schema (4)

- ▶ May mitigate problems of OOV entity pairs
- ▶ Characteristic combination of logits before applying sigmoid:

$$\theta_{e_s, r, e_o}^{FEN} = \theta_{e_s, r, e_o}^F + \theta_{e_s, r, e_o}^E + \theta_{e_s, r, e_o}^N$$

- ▶ Ranking loss as usual; if  $(e_s, r, e_o) \in \mathcal{O}$  and  $(e'_s, r, e'_o) \notin \mathcal{O}$ , want  $\theta_{e_s, r, e_o}^{FEN} > \theta_{e'_s, r, e'_o}^{FEN}$
- ▶ Term in objective to maximize is  $\log \left( \sigma(\theta_{e_s, r, e_o}^{FEN} - \theta_{e'_s, r, e'_o}^{FEN}) \right)$
- ▶ Why this particular objective?
- ▶ (More common to use ratio between probabilities, or difference between log probabilities)

## Universal schema for type prediction

- ▶ Avoid hand-designed type hierarchy (how many types are perfect, 5? 50? 500?)
- ▶ Avoid need for labeled data
- ▶ Types are the union of all available types from all sources
  - ▶ Multiple type hierarchies, possibly overlapping and redundant
  - ▶ Free-text expressions of types in corpus: “movie mogul James Cameron”, “Mohamed al-Fayed, a tycoon”
- ▶ Here each row corresponds to an entity (which could be textual, like “James Cameron”)
- ▶ Each column corresponds to a type, possibly textual
- ▶ (Type is expressed as a unary relation)
- ▶ Each entity  $e$  (possibly a free-text string) is associated with latent vector  $e$

## Universal schema for type prediction (2)

- ▶ Each type  $t$  (possibly a free-text string) is associated with latent vector  $\mathbf{t}$
- ▶ Models  $\Pr(e \in t) = \sigma(\mathbf{e} \cdot \mathbf{t})$
- ▶ New York Times, years 1990 to 2007
- ▶ NER tagging, dependency parsing
- ▶ Extract dependency paths originating from a (named) entity mention as unary relations
- ▶ Traverse from the head token of the entity mention to the root of the dependency tree
- ▶ Lexicalized path from the entity mention to content word (noun, adjective) used as one unary relation
- ▶ Stop when approaching a verb or a clause boundary
- ▶ Resulting matrix has 500k rows, 17k columns

## Universal schema for type prediction (3)

Entity	Observed	Predicted
Mohamed al-Fayed	tycoon, owner, entrepreneur, financier, estate of X, purchase by X	billionaire, millionaire, magnate, person:Freebase
House of Pain	Xs member, reminiscent of X, music by X, rap group X	trio, band, X rap, singer of X, rapper X
Sprite	commercial, drink, brand, ad, drink X, campaign for X	product:Freebase, food:Freebase
Pierre Louys	poem of, novel by, X's poem, X's novel	poet, novelist, author
Rick Wamsley	goaltender, goalie	goalkeeper

## OpenIE: some milestones

- ▶ 1992, Hearst patterns [2]
- ▶ 2004, KnowItAll, entity-type [3]
- ▶ 2007, TextRunner, entity-relation [4]
- ▶ 2009, USP (unsupervised semantic parsing), uses Markov logic networks (MLN) [5]
- ▶ 2011, generative model [6]
- ▶ 2013, universal schema [1]

## References

- [1] S. Riedel, L. Yao, A. McCallum, and B. M. Marlin, "Relation extraction with matrix factorization and universal schemas," in *NAACL Conference*, 2013, pp. 74–84. [Online]. Available:  
<http://www.anthology.aclweb.org/N/N13/N13-1008.pdf>
- [2] M. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *International Conference on Computational Linguistics*, vol. 14, 1992, pp. 539–545. [Online]. Available:  
[http://www.aclweb.org/website/old\\_anthology/C/C92/C92-2082.pdf](http://www.aclweb.org/website/old_anthology/C/C92/C92-2082.pdf)
- [3] O. Etzioni, M. Cafarella *et al.*, "Web-scale information extraction in KnowItAll," in *WWW Conference*. New York: ACM, 2004. [Online]. Available: <http://www.cs.washington.edu/research/knowitall/papers/www-paper.pdf>
- [4] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open information extraction from the Web," in *IJCAI*, M. M. Veloso, Ed., 2007, pp. 2670–2676. [Online]. Available:  
<http://www.ijcai.org/Papers07/Papers/IJCAI07-429.pdf>

## References (2)

- [5] H. Poon and P. Domingos, "Unsupervised semantic parsing," in *EMNLP Conference*, 2009, pp. 1–10. [Online]. Available: <http://anthology.aclweb.org/D/D09/D09-1001.pdf>
- [6] L. Yao, A. Haghighi, S. Riedel, and A. McCallum, "Structured relation discovery using generative models," in *EMNLP Conference*, 2011, pp. 1456–1466. [Online]. Available: <http://anthology.aclweb.org/D/D11/D11-1135.pdf>
- [7] T. Mitchell *et al.*, "Never-ending learning," in *AAAI Conference*, 2015. [Online]. Available: [http://www.cs.cmu.edu/~tom/pubs/NELL\\_aaai15.pdf](http://www.cs.cmu.edu/~tom/pubs/NELL_aaai15.pdf)
- [8] Mausam, M. Schmitz, R. Bart, S. Soderland, and O. Etzioni, "Open language learning for information extraction," in *EMNLP*, 2012. [Online]. Available: <https://homes.cs.washington.edu/~mausam/papers/emnlp12a.pdf>
- [9] S. Saha, H. Pal *et al.*, "Bootstrapping for numerical open ie," in *ACL Conference*, vol. 2, 2017, pp. 317–323.

## References (3)

- [10] O. Etzioni, A. Fader, J. Christensen, S. Soderland, and M. Mausam, "Open information extraction: The second generation," in *IJCAI*, 2011, pp. 3–10. [Online]. Available: <https://www.aaai.org/ocs/index.php/IJCAI/IJCAI11/paper/viewFile/3353/3408>

## Continuous knowledge representation

## Continuous knowledge representation

- ▶ The goal is to embed not only words but also entities, types and relations
- ▶ Based on mostly-sound but largely-incomplete knowledge graphs (KGs)
- ▶ And unstructured corpora, perhaps annotated with entities and fine types
- ▶ The embeddings can make various tasks easier or better
- ▶ E.g., knowledge base completion (KBC), more accurate or extended fine type tagging or entity linking, inferring paraphrasing, entailment, or contradiction, etc.

# Tasks and datasets for KG/KB embedding

**WN18:** 41k entities (WordNet synsets), 18 relation types (hypernymy, synonymy, . . . ), folds 141k/5k/5k

**FB15k:** 15k Freebase entities, 1345 relation types, folds 483k/50k/59k

**FbS snapshots:** Two snapshots of Freebase some time interval apart; removes sampling idiosyncrasies

**NYT+FB:** Freebase triples, plus dependency path-based textual relations from New York Times; entity mentions aligned with FB entity IDs; 25k entities, 4k relation types

**FB15k+ClueWeb12:** Corpus is ClueWeb12 with Google entity annotations

- 
- ▶ WN18, FB15k, FbS snapshots used for knowledge base completion (KBC)
  - ▶ Algorithm fits model using train and dev folds; ranks all other triples; those in eval fold are “relevant” docs; use ranking performance measures like MAP, MRR
  - ▶ NYT+FB can be used for jointly embedding entities and relations informed by both KG and text

## Structured embedding (SE)

- ▶ Entity can act as subject or object
- ▶ Accordingly each entity  $e$  gets two vectors  $\vec{e}_s, \vec{e}_o$
- ▶ Each relation  $r$  is also associated with two matrices  $M_{rs}, M_{ro}$
- ▶ If  $M_{rs}\vec{e}_s$  and  $M_{ro}\vec{e}_o$  are “close”, then  $(e_s, r, e_o)$  is more likely, e.g.,

$$\Pr(e_s, r, e_o) = \sigma((M_{rs}\vec{e}_s) \cdot (M_{ro}\vec{e}_o))$$

- ▶ Or some decreasing function of  $\|M_{rs}\vec{e}_s - M_{ro}\vec{e}_o\|$
- ▶ In words, each relation has two associated projections that bring participating entities close after projection
- ▶ Negative sampling: if  $(e_s, r, e_o)$  holds, replace with (uniformly?) randomly sampled  $(e'_s, r, e'_o)$  and assume these do not hold
- ▶ (Exactly one perturbed in each negative instance, not both)

## Structured embedding (SE) (2)

- ▶ We want  $\|M_{rs}\vec{e}_s - M_{ro}\vec{e}_o\| \ll \|M_{rs}\vec{e}'_s - M_{ro}\vec{e}'_o\|$
- ▶ Let  $\gamma > 0$  be a margin hyperparameter
- ▶ Loss is defined as

$$\sum_{(e_s, e_o)} \sum_{(e'_s, e'_o)} \max \left\{ 0, \|M_{rs}\vec{e}_s - M_{ro}\vec{e}_o\| + \gamma - \|M_{rs}\vec{e}'_s - M_{ro}\vec{e}'_o\| \right\}$$

- ▶ What would happen if no margin were used ( $\gamma = 0$ )?
- ▶ Nonconvex; use gradient descent

## TransE

- ▶ Each entity  $e$  associated with one vector  $\vec{e}$
- ▶ Each relation  $r$  associated with one vector  $\vec{r}$
- ▶ Model: if  $(e_s, r, e_o)$  holds, expect  $\vec{e}_s + \vec{r} \approx \vec{e}_o$
- ▶ As with SE, perturb to  $e'_s, e'_o$  and expect  
$$\|\vec{e}_s + \vec{r} - \vec{e}_o\| \ll \|\vec{e}'_s + \vec{r} - \vec{e}'_o\|$$
- ▶ And therefore the loss to minimize is

$$\sum_{(e_s, e_o)} \sum_{(e'_s, e'_o)} \max \left\{ 0, \gamma + \|\vec{e}_s + \vec{r} - \vec{e}_o\| - \|\vec{e}'_s + \vec{r} - \vec{e}'_o\| \right\}$$

- ▶ Fewer parameters than SE
- ▶ Cannot model many-to-one, one-to-many, or many-to-many relations

## TransR and STransE

- ▶ Mixes up projection and translation
- ▶ In **TransR**, a single projection matrix  $M_r$  is associated with each  $r$ , and applied to both  $\vec{e}_s$  and  $\vec{e}_o$
- ▶ Each relation  $r$  also associated with translation  $\vec{v}_r$
- ▶ We expect  $M_r \vec{e}_s + \vec{v}_r \approx M_r \vec{e}_o$  if  $(e_s, r, e_o)$  is valid in the KG
- ▶ **STransE** keeps  $M_{rs}$  distinct from  $M_{ro}$  as in SE, but retains the translation  $\vec{v}_r$  of TransR
- ▶ I.e., we expect  $M_{rs} \vec{e}_s + \vec{v}_r \approx M_{ro} \vec{e}_o$  if  $(e_s, r, e_o)$  is valid in the KG
- ▶ Loss and training similar to SE and TransE

## TransH

- ▶ Each relation  $r$  associated with
  - ▶ A **hyperplane** defined by unit normal vector  $p_r$
  - ▶ A **translation** vector  $d_r$  as in TransE
- ▶ Project  $e_s$  on to hyperplane (" $e_s \downarrow p_r$ "), translate by  $d_r$ , compare with  $e_o \downarrow p_r$
- ▶ Consider  $e_s, e_o, p_r$  with their tails at the origin
- ▶  $e_s \downarrow p_r = e_s - (e_s \cdot p_r)e_s$ , likewise for  $e_o$
- ▶  $\Delta(e_s, r, e_o) = \|(e_s \downarrow p_r) + d_r - (e_o \downarrow p_r)\|_2$
- ▶ Again, do pairwise training via perturbation: expect  
 $\Delta(e_s, r, e_o) \ll \Delta(e'_s, r, e'_o)$

## ITransF [1]

- ▶ STransE uses too many parameters per relation which is not good for rare relations
- ▶ ITransF allows parameter sharing among relations by using a set of underlying "concept" matrices stacked as tensor  $D$
- ▶ Each relation  $r$  is associated with two attention vectors  $\alpha_{rs}$  and  $\alpha_{ro}$
- ▶  $M_{rs} = \alpha_{rs}D$  and  $M_{ro} = \alpha_{ro}D$
- ▶ We expect  $M_{rs}e_s + r \approx M_{ro}e_o$  for valid triples in KG
- ▶ Loss and training similar to STransE

## CP and Rescal decompositions

- ▶ Suppose  $e_s, e_o$  are column vectors
- ▶ Form outer product matrix  $e_s e_o^\top \in \mathbb{R}^{D \times D}$
- ▶ Makes explicit feature crosses
- ▶ Relation  $r$  represented by  $M_r \in \mathbb{R}^{D \times D}$
- ▶ Confidence that  $(e_s, r, e_o)$  holds in KG is large if  $M_r \bullet (e_s e_o^\top)$  is large, and vice versa, where  $\bullet$  is elementwise dot-product

$$\sum_{i,j} M_r[i, j] e_s[i] e_o[j] = e_s^\top M_r e_o$$

- ▶ In general  $e_s^\top M_r e_o \neq e_o^\top M_r e_s$ , so asymmetry can be supported
- ▶ Some systems prefer a diagonal (therefore symmetric) matrix for  $M_r$  to reduce trainable weights ... **DistMult**
- ▶ If we stack up the  $M_r$ s over all relations  $r$ , we get a 3-axis tensor  $\mathbf{M} \in \mathbb{R}^{D \times D \times R}$
- ▶ If  $r, r'$  hold (and not hold) frequently between  $e_s, e_o$ , their “plates” in  $\mathbf{M}$  should be similar

## CP and Rescal decompositions (2)

- ▶ The 2-axis regular matrix analog would be, some rows (or columns) are similar to, or linearly dependent on, other rows (or columns)
- ▶ In which case, it would have low(er than full) rank
- ▶ Factorization via SVD reveals rank of a 2-axis matrix
- ▶ Defining and estimating rank of a tensor are more tricky
- ▶ **Candecomp/Parafac** is one approximate decomposition scheme:

$$\mathbf{X} \approx \sum_{i=1}^I \mathbf{a}_i \otimes \mathbf{b}_i \otimes \mathbf{c}_i$$

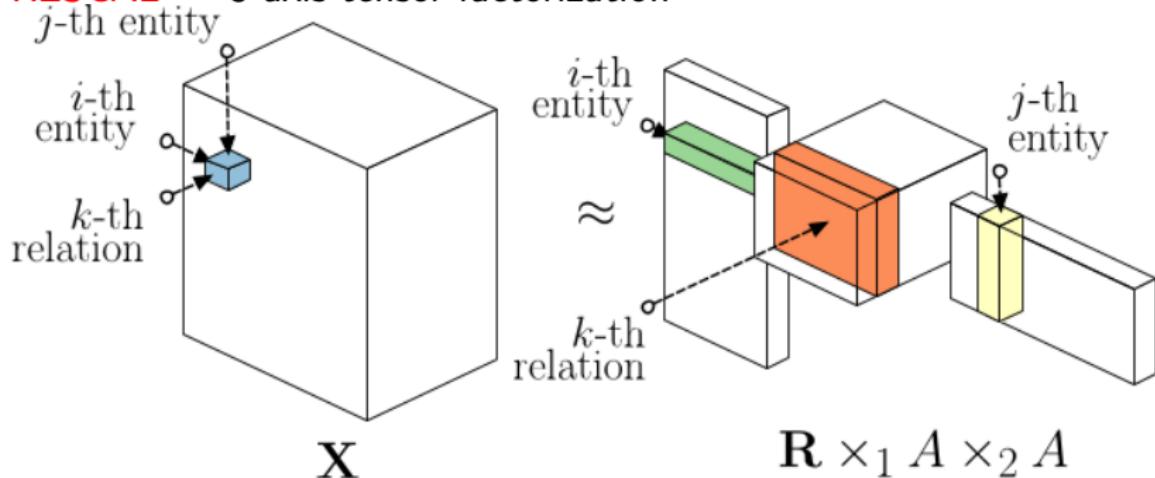
i.e.,

$$X[\ell, m, n] = \sum_i a_i[\ell] b_i[m] c_i[n]$$

where  $\mathbf{X} \in \mathbb{R}^{L \times M \times N}$  and  $\mathbf{a}_i \in \mathbb{R}^L, \mathbf{b}_i \in \mathbb{R}^M, \mathbf{c}_i \in \mathbb{R}^N$  and  $I$  is our control on the “rank” of approximating  $\mathbf{X}$

## CP and Rescal decompositions (3)

- **RESCAL** — 3-axis tensor factorization



- Assume  $N$  entities,  $R$  relations,  $D$  is the embedding dimension
- In tensor notation,

$$\mathbf{X} \approx \mathbf{M} \times_1 A \times_2 A$$

where  $\mathbf{X} \in \mathbb{R}^{N \times N \times R}$ ,  $\mathbf{M} \in \mathbb{R}^{D \times D \times R}$ ,  $A \in \mathbb{R}^{N \times D}$

## CP and Rescal decompositions (4)

- ▶ For the  $r$ th plate,

$$X[r, :, :] \approx A^\top M[r, :, :] A$$

- ▶ Overall loss objective to minimize is

$$\underset{A, M}{\operatorname{argmin}} \|X - M \times_1 A \times_2 A\|^2 + \lambda_A \|A\|^2 + \lambda_M \|M\|^2$$

- ▶ Overall non-convex, use alternating least-squares
  - ▶ Fix  $A$  and improve  $M$
  - ▶ Fix  $M$  and improve  $A$
- ▶ Can use SGD or batch solvers
- ▶ Quite a bit more compute-intensive than SVD/NMF style matrix decomposition

## Squashing the tensor into a matrix

- ▶ Each row represents a **pair** of entities  $(e_s, e_o)$
- ▶ Each column represents a relation  $r$
- ▶  $X[(e_s, e_o), r] = 1$  if the  $(e_s, r, e_o)$  holds in KG
- ▶ If two relations are very similar, their columns will be similar in  $X$
- ▶ Now do a standard SVD while controlling rank, giving  $D \ll N, D \ll R$  dimensional embeddings for each relation and entity pair
- ▶ Called “**matrix factorization**” (MF) approach or “F” model
- ▶ Downsides
  - ▶ Out-of-vocabulary (OOV) entity pairs at test time
  - ▶ No per-entity intrinsic representation

## Gaussian embeddings

- ▶ Bach was a famous classical composer and a man
- ▶ But not all men or composers are Bach
- ▶ If Bach and man are both represented using single vectors, must model and predict asymmetric containment relation between them
- ▶ One approach is to model each word  $i$  not with a single vector but a Gaussian density  $\mathcal{N}(\mu_i, \Sigma_i)$
- ▶ Word embeddings are usually compared using vector inner product
- ▶ Extend to distributions:

$$\begin{aligned}\langle (\mu_i, \Sigma_i), (\mu_j, \Sigma_j) \rangle &= \int_x \mathcal{N}(x|\mu_i, \Sigma_i) \mathcal{N}(x|\mu_j, \Sigma_j) dx \\ &= \mathcal{N}(\vec{0}|\mu_i - \mu_j, \Sigma_i + \Sigma_j)\end{aligned}$$

## Gaussian embeddings (2)

- ▶ Asymmetric containment captured via KL divergence
- ▶ If  $y$  is contained in  $x$ , then the KL divergence from  $x$  to  $y$  is small
- ▶  $D_{\text{KL}}(\mathcal{N}_j \parallel \mathcal{N}_i)$  has a closed form wrt  $\mu_i, \Sigma_i, \mu_j, \Sigma_j$
- ▶ Train with a margin and hinge loss
  - ▶ WordNet has aliases for each synsets  $i \equiv j$ : encourage large inner product between these
  - ▶ Randomly perturb to get negative pairs  $i \not\equiv j$ : encourage small inner product between these
  - ▶ Use WordNet hypo/hypernym instances  $i \subseteq j$ : encourage small KL distance
  - ▶ Randomly perturb to get negative instances  $i \not\subseteq j$  — encourage large KL distance

## Order embeddings

- ▶ Specialized to represent partial orders like  $e \in t$  and  $t_1 \subseteq t_2$ , denoted uniformly as  $x_1 \prec x_2$
- ▶ Embed each  $x$  to vector  $\mathbf{x}$
- ▶ If  $x_1 \prec x_2$ , assert  $\mathbf{x}_1 \leq \mathbf{x}_2$ , elementwise
- ▶ E.g., by assessing a hinge loss  $\|(\mathbf{x}_1 - \mathbf{x}_2)_+\|_1$ , where  $(\mathbf{a})_+ = (\max\{0, a_i\})$  is an elementwise ReLU
- ▶ Negative sampling as usual: pick  $x_1 \prec x_2$ , perturb either to get  $x'_1 \not\prec x'_2$ ; no checking for false negative
- ▶ Let  $E(x_1, x_2) = \|(\mathbf{x}_1 - \mathbf{x}_2)_+\|_1$
- ▶ Loss function is:

$$\sum_{x_1 \prec x_2} E(x_1, x_2) + \sum_{x_1 \not\prec x_2} \max \{0, \gamma - E(x_1, x_2)\}$$

where  $\gamma$  is a margin

## Order embeddings (2)

- ▶ Used to model hypernymy in WordNet [2]
- ▶ 70k noun hypernymy pairs  $(x_1, x_2)$ ; transitive closure leads to 800k pairs; 4000 sampled as test fold, rest used for training

Method	Accuracy
Transitive closure	88.2
Gaussian embeddings	86.6
Order embeddings	90.6

- ▶ Not surprising transitive closure is so good
- ▶ Reducing training fold size results in step drop in performance of transitive closure, but OE is significantly less damaged
- ▶ Negative sample makes major difference

## HoIE : Holographic Embedding [3]

- ▶ Entities and relations are modeled as vectors similar to DistMult
- ▶ Model is motivated by holographic models of associative memory and it learns compatibility between relations and entity pairs
- ▶ Confidence of correctness of a triple is proportional to  $r^\top(e_s \star e_o)$  where  $e_s \star e_o$  represents the circular correlation between vectors  $e_s$  and  $e_o$  and

$$[e_s \star e_o]_k = \sum_{i=0}^{n-1} e_{s_i} e_{o_{(k+i) \bmod n}}$$

- ▶ Circular correlation is asymmetric, thus HoIE allows asymmetric relations
- ▶ Logistic loss or pair-wise ranking loss can be used for training

## ComplEx [4]

- ▶ Entities and relations are modeled as vectors in Complex domain
- ▶ Confidence of correctness of a triple is proportional to the complex dot product between  $e_s$  and  $e_o$  weighted by  $r$

$$Pr(e_s, r, e_o) = \sigma(Re(\langle r, e_s, \overline{e_o} \rangle))$$

- ▶ Similar to DistMult but in Complex domain
- ▶ Allows handling symmetric, asymmetric and anti-symmetric relations together
- ▶ Logistic loss for training

## The menagerie

Model	Score function $s(e_s, r, e_o)$	
SE	$\ M_{rs}e_s - M_{ro}e_o\ _2$	
TransE	$\ e_s + v_r - e_o\ _2$	
STransE	$\ M_{rs}e_s + v_r - M_{ro}e_o\ _2$	
TransR	$\ M_r e_s + v_r - M_r e_o\ _2$	$M_{rs} = M_{ro} = M_r$
DistMult	(Maximize) $e_s^\top M_r e_o$	$M_r$ diagonal
TransH	$\ e_s - (e_s \cdot p_r)e_s + d_r - e_o + (e_o \cdot p_r)e_o\ _2$	$\ p_r\ _2 = 1$
TransD	$\ (p_r e_s^\top + \mathbb{I})e_s + d_r - (p_r e_o^\top + \mathbb{I})e_o\ _2$	
NTN	$u_r^\top \tanh(e_s^\top \mathbf{M}_r e_o + M_{rs}e_s + M_{ro}e_o + b_r)$	$\mathbf{M}_r \in \mathbb{R}^{D \times D \times K}$

- ▶ Who is better than who?
- ▶ Just check the publication date!

## Performance on Link Prediction

Method	WN18			FB15k		
	MR	MRR	H10	MR	MRR	H10
ITransF	223		<b>95.2</b>	77		81.4
ComplEx		<b>94.1</b>	94.7		<b>69.2</b>	<b>84</b>
STransE	244		94.7	69		79.7
HoIE		<b>93.8</b>	94.9		52.4	73.9
TransR	225		92	77		68.7
TransH	303		86.7	87		64.4
DistMult		82.2	93.6		65.4	82.4
TransE	251	45.4	93.4	125	38.0	47.1

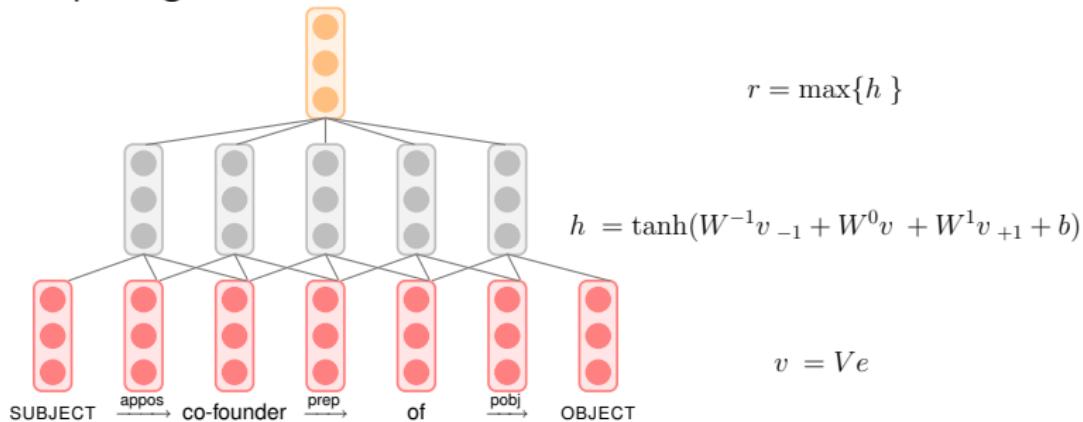
# Embeddings from combining KG and text

- ▶ TL;DR Use convnet on dependency path to get compositional embedding for  $r$ , then combine using DistMult with entity embeddings
- ▶ Example dependency paths for (person, founded, organization)

Textual Pattern	Count
SUBJECT $\xrightarrow{\text{appos}}$ founder $\xrightarrow{\text{prep}}$ $\xrightarrow{\text{of}}$ $\xrightarrow{\text{pobj}}$ OBJECT	12
SUBJECT $\xleftarrow{\text{nsubj}}$ co-founded $\xrightarrow{\text{dobj}}$ OBJECT	3
SUBJECT $\xrightarrow{\text{appos}}$ co-founder $\xrightarrow{\text{prep}}$ $\xrightarrow{\text{of}}$ $\xrightarrow{\text{pobj}}$ OBJECT	3
SUBJECT $\xrightarrow{\text{conj}}$ co-founder $\xrightarrow{\text{prep}}$ $\xrightarrow{\text{of}}$ $\xrightarrow{\text{pobj}}$ OBJECT	3
SUBJECT $\xleftarrow{\text{pobj}}$ with $\xrightarrow{\text{prep}}$ co-founded $\xrightarrow{\text{dobj}}$ OBJECT	2
SUBJECT $\xleftarrow{\text{nsubj}}$ signed $\xrightarrow{\text{xcomp}}$ establishing $\xrightarrow{\text{dobj}}$ OBJECT	2
SUBJECT $\xleftarrow{\text{pobj}}$ with $\xrightarrow{\text{prep}}$ founders $\xrightarrow{\text{prep}}$ $\xrightarrow{\text{of}}$ $\xrightarrow{\text{pobj}}$ OBJECT	2
SUBJECT $\xrightarrow{\text{appos}}$ founders $\xrightarrow{\text{prep}}$ $\xrightarrow{\text{of}}$ $\xrightarrow{\text{pobj}}$ OBJECT	2
SUBJECT $\xleftarrow{\text{nsubj}}$ one $\xrightarrow{\text{prep}}$ $\xrightarrow{\text{of}}$ $\xrightarrow{\text{pobj}}$ founders $\xrightarrow{\text{prep}}$ $\xrightarrow{\text{of}}$ $\xrightarrow{\text{pobj}}$ OBJECT	2
SUBJECT $\xleftarrow{\text{nsubj}}$ founded $\xrightarrow{\text{dobj}}$ production $\xrightarrow{\text{conj}}$ OBJECT	2
SUBJECT $\xleftarrow{\text{appos}}$ partner $\xrightarrow{\text{pobj}}$ with $\xrightarrow{\text{prep}}$ founded $\xrightarrow{\text{dobj}}$ production $\xrightarrow{\text{conj}}$ OBJECT	2
SUBJECT $\xleftarrow{\text{pobj}}$ by $\xrightarrow{\text{prep}}$ co-founded $\xrightarrow{\text{rcmod}}$ OBJECT	1
SUBJECT $\xleftarrow{\text{nn}}$ co-founder $\xrightarrow{\text{prep}}$ $\xrightarrow{\text{of}}$ $\xrightarrow{\text{pobj}}$ OBJECT	1
SUBJECT $\xrightarrow{\text{dep}}$ co-founder $\xrightarrow{\text{prep}}$ $\xrightarrow{\text{of}}$ $\xrightarrow{\text{pobj}}$ OBJECT	1
SUBJECT $\xleftarrow{\text{nsubj}}$ helped $\xrightarrow{\text{xcomp}}$ establish $\xrightarrow{\text{dobj}}$ OBJECT	1
SUBJECT $\xleftarrow{\text{nsubj}}$ signed $\xrightarrow{\text{xcomp}}$ creating $\xrightarrow{\text{dobj}}$ OBJECT	1

## Embeddings from combining KG and text (2)

- ▶ Composing text into  $M$  of DistMult



- ▶ (Presumably LSTMs have been tried too)
- ▶ Model  $p(e_o|e_s, r; \Theta) = \frac{\exp(f(e_s, r, e_o; \Theta))}{\sum_{e' \in \text{Neg}(e_s, r, ?)} \exp(f(e_s, r, e'; \Theta))}$
- ▶  $f$  is the function implemented by the convnets and a final DistMult:  $\vec{e}_s^\top \text{diag}(r) \vec{e}_o$

## Embeddings from combining KG and text (3)

- ▶ Two potential issues
  - ▶ Although Neg is sampled from all possible negative  $e'$ , denominator not scaled suitably
  - ▶ Positive term not included in denominator
- ▶  $\Theta$  includes  $M_r$  from KG, convnet weights from corpus, and  $\vec{e}$
- ▶ Overall objective to maximize for each source is

$$L(\mathcal{T}; \Theta) = \sum_{(e_s, r, e_o) \in \mathcal{T}} \log p(e_o | e_s, r; \Theta) + \sum_{(e_s, r, e_o) \in \mathcal{T}} \log p(e_s | e_o, r; \Theta)$$

where  $\mathcal{T} \in \{\mathcal{T}_{\text{KG}}, \mathcal{T}_{\text{corpus}}\}$

- ▶ Global objective to maximize is

$$L(\mathcal{T}_{\text{KG}}; \Theta) + \clubsuit L(\mathcal{T}_{\text{corpus}}; \Theta) - \spadesuit \|\Theta\|^2$$

- ▶ Is  $M_r$  updated only via  $\vec{e}$ s or directly from convnets?

## References

- [1] Q. Xie, X. Ma, Z. Dai, and E. Hovy, "An interpretable knowledge transfer model for knowledge base completion," *arXiv preprint arXiv:1704.05908*, 2017. [Online]. Available: <https://arxiv.org/pdf/1704.05908.pdf>
- [2] I. Vendrov, R. Kiros, S. Fidler, and R. Urtasun, "Order-embeddings of images and language," *arXiv preprint arXiv:1511.06361*, 2015. [Online]. Available: <https://arxiv.org/pdf/1511.06361>
- [3] M. Nickel, L. Rosasco, T. A. Poggio *et al.*, "Holographic embeddings of knowledge graphs," in *AAAI Conference*, 2016, pp. 1955–1961. [Online]. Available: <https://arxiv.org/abs/1510.04935>
- [4] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *ICML*, 2016, pp. 2071–2080. [Online]. Available: <http://arxiv.org/abs/1606.06357>
- [5] A. Bordes, J. Weston, R. Collobert, and Y. Bengio, "Learning structured embeddings of knowledge bases," in *AAAI Conference*, 2011, pp. 301–306. [Online]. Available: <http://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/viewFile/3659/3898>

## References (2)

- [6] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *NIPS Conference*, 2013, pp. 2787–2795. [Online]. Available: <http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data.pdf>
- [7] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix." in *ACL Conference*, 2015, pp. 687–696. [Online]. Available: <http://www.aclweb.org/anthology/P/P15/P15-1067.pdf>
- [8] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," in *EMNLP Conference*, 2015, pp. 1499–1509. [Online]. Available: <https://www.aclweb.org/anthology/D/D15/D15-1174.pdf>

## Querying KG and annotated corpus

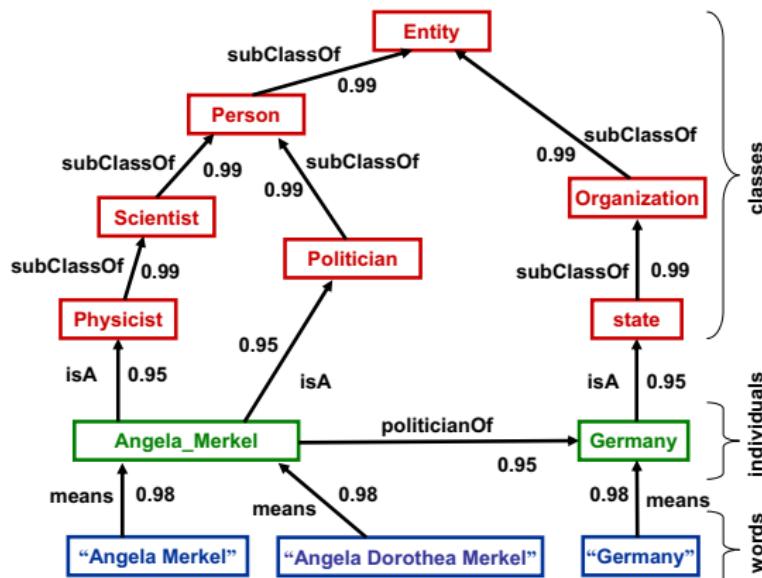
## Paradigms of KG and corpus search

- ▶ Assuming information sources are consolidated into a KG, and queries are provided in a structured graphical form, retrieve and rank response subgraphs [13]
- ▶ Compile well-formed question into a structured graph interpretation of a query, then execute it on KG [14, 15, 16]
- ▶ Annotate corpus with entity and type mentions, then rank entities in response to weakly structured keyword queries [10, 17]
- ▶ Combine KG and corpus to jointly rank entities in response to well-formed or keyword queries [18, 19, 20]

## Search overview

- ▶ Text search in RDBMS
  - ▶ WHIRL [21], DBXplorer [7], DISCOVER [22], BANKS [1]
- ▶ Text search in graph data
  - ▶ Text+XML: ELIXIR [5], XIRQL [6], BANKS, NAGA [12]
- ▶ Proximity search
  - ▶ OBJECTRANK [8], HUBRANK [23], NETRANK [24]
- ▶ Bridging the structured-unstructured divide
  - ▶ IR4QA [10], ENTITYRANK [9], QCQ [25]
- ▶ Question answering
  - ▶ Semantic query interpretation [14, 26, 15]
  - ▶ Joint query interpretation and response ranking [27, 18]

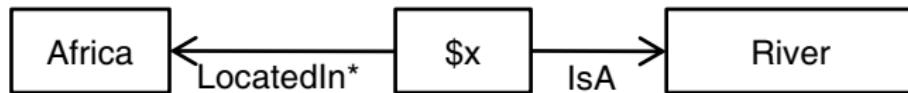
# NAGA data fragment



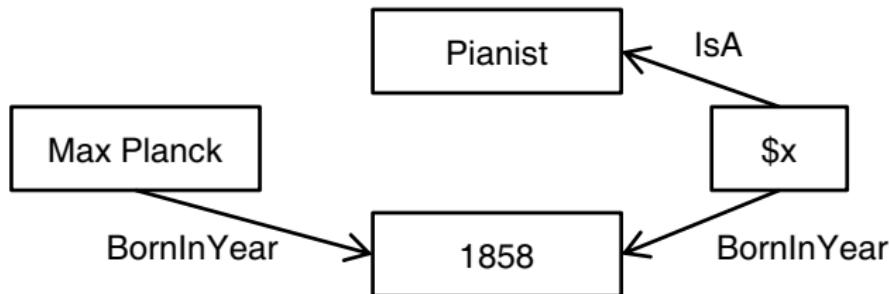
- ▶ “Means”, “is-a” from NE tagging, disambiguation
- ▶ “Subclass of” from WordNet, Wikipedia
- ▶ “Politician of” from relation extraction, Wikipedia
- ▶ Edge scores derived from **trustworthiness** of source page and **accuracy/confidence** of extraction

## Sample NAGA queries

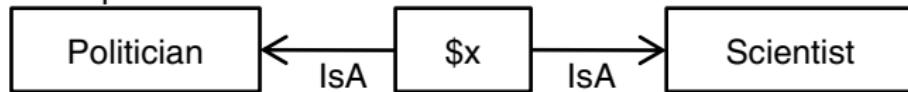
River located in Africa



Pianist born in the same year as Max Planck



Both politician and scientist



## Answer graph search and scoring overview

- ▶ Each edge in query graph is a **fact template**
- ▶ Some nodes are **bound** to constants, others like `?x` are **variables** to be instantiated
- ▶ Edges also generally have labels or regular expressions to be matched in data graph
- ▶ An **answer graph** is a subgraph of KG that satisfies all label constraints on nodes and edges of the query graph, and binds all variables to ground values in KG
- ▶ Multiple answer graphs may be found, in which case they must be scored and ranked
  - ▶ Prefer **confident** answers where all KG claims have large probability
  - ▶ Prefer **informative/specific** answers: Einstein is-a **physicist** better than **human**
  - ▶ Prefer **compact** connections, e.g., how are Einstein and Bohr related?
    - ▶ Einstein and Tom Cruise are/were both vegetarians
    - ▶ Tom Cruise was born the year Bohr died

## Fact template and answer graph

- ▶ A **query** is a connected directed graph  $Q = (V, E, L_V, L_E, U)$
- ▶  $V$  is a set of vertices
- ▶  $E \subset V \times V$  is a set of edges
- ▶  $L_V$  is a set of node labels
- ▶  $L_E$  is a set of relation (edge) labels
- ▶  $U$  is a set of variables
- ▶ Each vertex  $v \in V$  is assigned a label  $\ell(v) \in L_V \cup U$
- ▶ Each edge  $e \in E$  is assigned a label  
 $\ell(e) \in \text{REGEx}(L_E) \cup U \cup \{\text{CONNECT}\}$
- ▶ If an edge or vertex is labeled with a variable, we call that edge or vertex unbound
- ▶ We disallow unbound edges between two unbound vertices
- ▶ Edge of query graph is called a **fact template**

## Fact template and answer graph (2)

- ▶ **Matching path** for a fact template  $(x, r, y)$  is a sequence of edges  $m_1 \dots m_n$  in the KG such that:
  - ▶ If  $r$  is a variable then  $n = 1$  and endpoints of  $m_1$  match  $x$  and  $y$
  - ▶ If  $r$  is a REGEx then  $\ell(m_1) \dots \ell(m_n)$  is in the regular language defined by  $r$
  - ▶ If  $r = \text{CONNECT}$  then  $m_1 \dots m_n$  is an undirected path with endpoints matching  $x, y$
- ▶ **Answer graph**  $g$  is a subgraph of KG such that
  - ▶ Each fact template of  $q$  has a matching path in the answer graph
  - ▶ Each fact in  $g$  is part of one matching path
  - ▶ Each vertex of  $q$  is bound to exactly one vertex of  $g$
  - ▶ ( $g$  may include additional vertices)

## NAGA answer graph ranking

- ▶ Prefer **confident** answers, where quality of fact extraction is high
- ▶ Prefer **informative** answers: Prefer “Albert Einstein” is-a **physicist** to “Albert Einstein” is-a **scientist**
- ▶ Prefer **compact** answers (as in BANKS)
  - ▶ How are Einstein and Bohr related?
  - ▶ Einstein and Tom Cruise were/are vegetarians
  - ▶ Tom Cruise was born the year Bohr died
- ▶ Query  $q = q_1 \cdots q_n$ , candidate answer graph  $g = g_1 \cdots g_n$
- ▶ Each  $q_i$  is a fact template,  $g_i$  is a grounded fact
- ▶ Want to model  $\Pr(q|g) = \prod_i \Pr(q_i|g)$
- ▶ Similar to probabilistic information retrieval (PIR) using language models

## NAGA answer graph ranking (2)

- ▶  $g$  is like a “document” in PIR

$$\Pr(q_i|g) = \alpha \tilde{\Pr}(q_i|g) + (1 - \alpha) \tilde{\Pr}(q_i|\text{KG}),$$

where  $\alpha \in (0, 1)$  is a blender between generating  $q_i$  from  $g$  vs. whole KG

- ▶ Next we model

$$\tilde{\Pr}(q_i|g) = \beta \Pr_{\text{conf}}(q_i|g) + (1 - \beta) \Pr_{\text{info}}(q_i|g), \quad \beta \in (0, 1)$$

- ▶ We can be confident that Einstein was both a Physicist and politician, but he was much more of a Physicist

$$\Pr_{\text{info}}(q_i|g) = \prod_{f \in \text{match}(q_i, g)} \Pr_{\text{finfo}}(f|q_i)$$

- ▶ Fact informativeness (finfo) depends on supporting witness

## NAGA answer graph ranking (3)

- Given query template  $(x', r', y')$  and fact  $f = (x, r, y)$ ,

$$\Pr_{\text{finfo}}(f|q_i) = \begin{cases} \Pr(x|r, y), & \text{if } x' \text{ is unbound in } q_i \\ \Pr(y|r, x), & \text{if } y' \text{ is unbound in } q_i \\ \Pr(r|x, y), & \text{if } r' \text{ is unbound in } q_i \\ \Pr(x, y|r), & \text{if } x', y' \text{ is unbound in } q_i \\ \Pr(x, r|y), & \text{if } x', r' \text{ is unbound in } q_i \\ \Pr(r, y|x), & \text{if } r', y' \text{ is unbound in } q_i \\ \Pr(x, r, y), & \text{otherwise} \end{cases}$$

- Empirically approximate

$$\Pr(x, r, y) \approx \frac{|W(x, r, y)|}{\sum_{x', r', y'} |W(x', r', y')|}$$

where  $W(\dots)$  is the number of witness (pages)

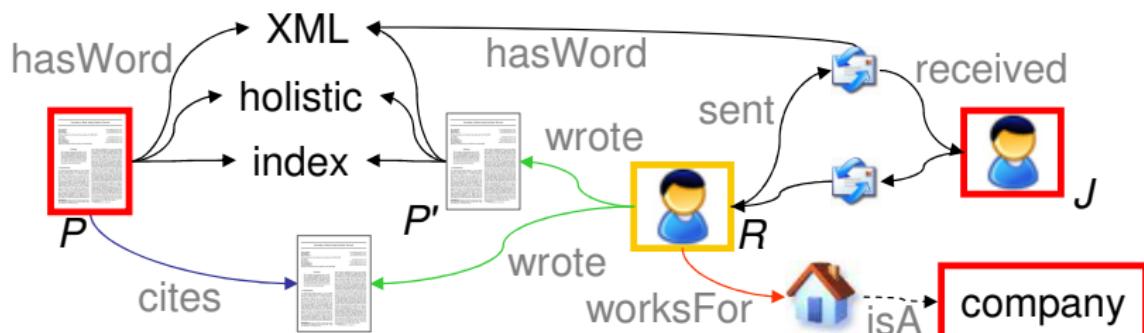
## NAGA answer graph ranking (4)

- ▶ Demo
- ▶ Much followup work after NAGA; some well-known samples:
  - ▶ Efficient subgraph matching on billion node graphs [28]
  - ▶ Learning to create data-integrating queries [29]
  - ▶ Keyword Search on External Memory Data Graphs [30]
  - ▶ BLINKS: ranked keyword searches on graphs [31]

# Graphs from desktop and enterprise search

Journal editor  $J$  must find reviewer  $R$  from a company for a submitted paper  $P$

- ▶  $P$  shares words with papers  $P'$  written by  $R$
- ▶  $P$  cites papers  $P'$  written by  $R$
- ▶  $R$  works for organization  $O$  is-a company
- ▶  $R$  and  $J$  have exchanged many emails



Proximity may be decided by multiple paths

## Typed graphs are ubiquitous

- ▶ Web search with diversified link types: same/different-domain, subdirectory, linked-from-blog, ...
- ▶ Entities mentioned on Web pages, entity is-a type, Web pages pertaining to topics, ...
- ▶ Collaborative search: person liked movie, movie of-genre drama, person tagged image with tag *Hockey*, image embedded-in page, image has caption word *meadow* ...
- ▶ XML and RDBMS-turned-into-graphs
- ▶ Confluence of structured and free-format, keyword-based search
- ▶ Many useful applications: product catalogs, software libraries, even Web search

## Definitions of graph proximity

Several choices for ranking single nodes  $v$  in response to query node  $u$

**Conductance/personalized PageRank:** In a random walk with restart at  $u$ , the steady-state probability of visiting  $v$

**Hitting time:**  $H(u, v)$  is the expected number of steps to reach  $v$  for the first time, starting from  $u$ .

**Commute time:**  $R(u, v)$  is the expected number of steps to go from  $u$  to  $v$  back to  $u$ ; by linearity of expectation,  
$$R(u, v) = H(u, v) + H(v, u).$$

**SimRank:** Similarity between query node  $u$  and candidate node  $v$  is (damped) average of (recursive) similarity of nodes  $a$  linking to  $u$  and  $b$  linking to  $v$

**Escape probability:** The probability that a random walk starting at  $u$  reaches  $v$  before returning to  $u$

## Proximity in entity-relationship graphs

- ▶ Node = entity (page), edge = relationship (hyperlink)
- ▶ Each edge  $(u, v)$  has **edge conductance**  
 $C(v, u) = C(u \rightarrow v) = \Pr(v|u)$
- ▶ Uniform for standard PageRank
- ▶ From each node, **teleport** with probability  $1 - \alpha$
- ▶ If teleporting, go to node  $u$  with probability  $r(u)$
- ▶ Overall PageRank equation  $p_r = \alpha C p_r + (1 - \alpha)r$
- ▶ Usually, given  $C, r, \alpha$ , find  $p_r$  offline

## Nice properties

Assume teleport  $\delta_u$  to a single node  $u$ , and consider score of  $v$

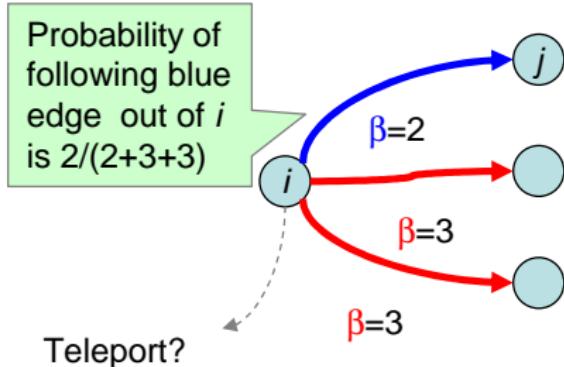
- ▶  $v$  has large score if there is/are **high-conductance** path/s from  $u$  to  $v$  ... shortest path
- ▶  $v$  has large score if there are **many** paths from  $u$  to  $v$  ... parallel paths
- ▶  $v$  has large score if paths from  $u$  to  $v$  do not branch off to many other nodes ... avoid “**postman hubs**”

## An inverse problem

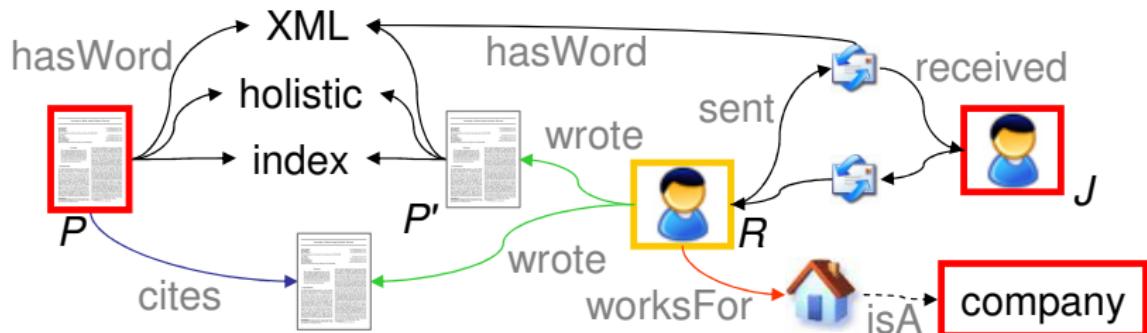
- ▶ Pair preference: " $i \prec j$ " means node  $i$  less preferred than node  $j$
- ▶ Underlying generative assumption: Hidden  $C$  generated  $p_r$ , and nodes ordered in decreasing  $p_r(v)$
- ▶ Pair preferences sampled from this total order, presented to algorithm
- ▶ (Training data may not follow assumption exactly; i.e., be noisy)
- ▶ Learner predicts  $u \prec v$  or  $u \succ v$  for test node pair
- ▶ Also of interest: can learner recover  $C$ ?

## Edge types determine conductance

- ▶ Nodes have entity types: Person, Paper, Email, Company
- ▶ Edges have relation types: wrote, sent, cited, in-reply-to
- ▶ Edge  $e$  has type  $t(e) \in \{1, \dots, T\}$
- ▶ Edge  $(u, v)$  of type  $t(u, v)$  has weight  $\beta(t(u, v))$  and conductance  $C(v, u)$



## Query and learning model



- ▶ Teleport vector  $r$  determined at query time
- ▶ Teleport into nodes that match query predicates
- ▶ May be updated dynamically by browsing results
- ▶ Given  $\prec$  we want to estimate  $\beta_1, \dots, \beta_T$ , thereby estimating  $C$
- ▶  $C$  cannot be too complex thanks to global constraints via  $\beta$

## Conductance matrix

- ▶ Create dummy node  $d$
- ▶ Create edges  $(d, u)$  and  $(u, d)$  for each  $u$
- ▶ Let  $0 < \alpha < 1$  be the teleport probability
- ▶ Let  $r_v$  be the probability of teleport to  $v$
- ▶ Assuming no dead end,

$$C_{\{\alpha, \beta\}}(v, u) = \begin{cases} \frac{\alpha \beta(t(u, v))}{\sum_{(u, w) \in E} \beta(t(u, w))}, & u \neq d, v \neq d \\ 1 - \alpha, & u \neq d, v = d \\ r_v, & u = d, v \neq d \\ 0, & \text{otherwise} \end{cases}$$

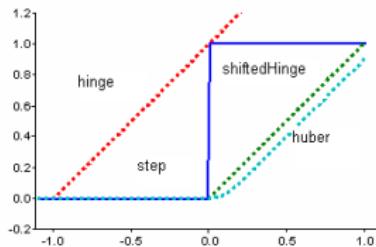
## NETRANK Optimization

- ▶  $C$  is a function of  $\beta$ , i.e.,  $C(\beta)$
- ▶ Scaling all  $\beta$  by any positive factor keeps all  $C(\cdot, \cdot)$  unchanged
- ▶ Find  $\beta \geq \vec{1}$  such that
  - ▶  $p = C(\beta)p$  ... quadratic constraints
  - ▶  $p_i \leq p_j$  for all  $i \prec j$
- ▶ Replace constraint  $p_i \leq p_j$  by a loss function  $\text{loss}(p_i - p_j)$ 
  - ▶  $p_i < p_j \implies$  zero or small loss
  - ▶  $p_i \geq p_j \implies$  large loss
- ▶ Margin? Want  $p_i + 1 \leq p_j$ 
  - ▶ Scaling  $\beta$  has no effect on satisfying margin
  - ▶ Works well without margin in practice

## Three problems to solve

Design SmoothLoss( $p_u - p_v$ )

$$\text{huber}(z) = \begin{cases} 0, & z < 0 \\ z^2/(2W), & z \in (0, W] \\ z - W/2, & z > W \end{cases}$$



Design ModelCost( $\beta$ )

- ▶  $\text{ModelCost}(\beta) = \sum_t (\beta(t) - 1)^2$ , or
- ▶  $\sum_{t,t'} (\beta(t) - \beta(t'))^2$

Avoid quadratic constraints  $p = C(\beta)p$

PageRank within Newton updates

## ModelCost( $\beta$ ): Parsimonious choice of $\beta$

- ▶ If  $\beta = \vec{1}$ , we get unweighted Pagerank
- ▶ Therefore a reasonable  $\text{ModelCost}(\beta) = \sum_t (\beta(t) - 1)^2$
- ▶ In fact, we get unweighted Pagerank if all  $\beta(t)$  are **equal**, not necessarily all equal to one
- ▶ Another choice is  $\sum_{t,t'} (\beta(t) - \beta(t'))^2$
- ▶ Discourages large multiplicative factors . . .  
 $\text{ModelCost}(K\beta) = K^2 \text{ModelCost}(\beta)$
- ▶ . . . but not additive terms:  
 $\text{ModelCost}(\beta + K\vec{1}) = \text{ModelCost}(\beta)$
- ▶ In practice these work about equally well

## Breaking the $p = C(\beta)p$ recursion

$$\min_{\substack{0 < \alpha < 1 \\ \beta \geq \mathbf{0}, p}} \text{ModelCost}(\beta) + B \sum_{i \prec j} \text{SmoothLoss}(p_u - p_v)$$

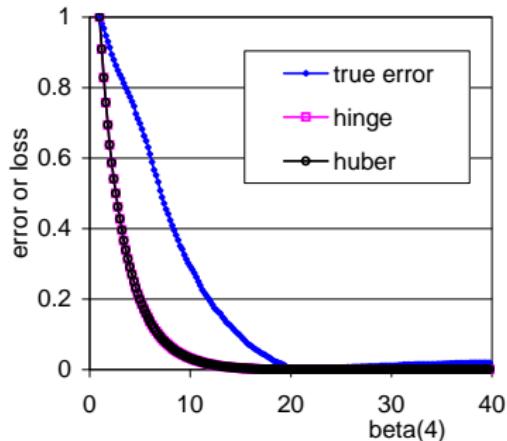
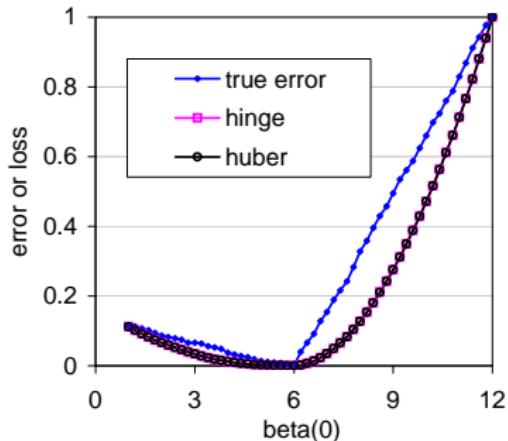
subject to 
$$p = C_{\{\alpha, \beta\}} p$$

Approximate  $p \approx C^H p_0$  where  $H$  is a (possibly adaptive) **horizon**

$$\begin{aligned} \min_{0 < \alpha < 1; \beta \geq \vec{1}} & \text{ModelCost}(\beta) + \\ & B \sum_{i \prec j} \text{SmoothLoss}\left((C^H p_0)_u - (C^H p_0)_v\right) \end{aligned}$$

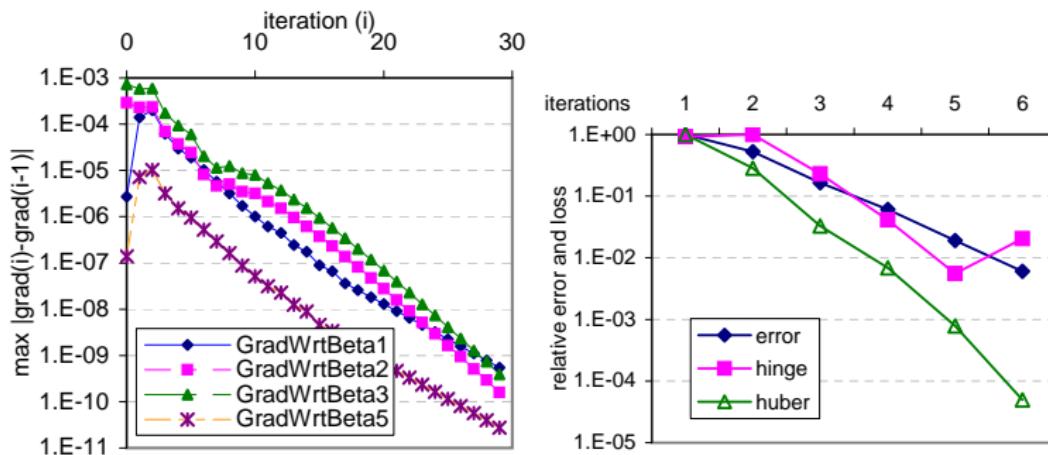
Can compute gradient wrt  $\beta$  similar to PageRank

# 0/1 loss and NETRANK approximations



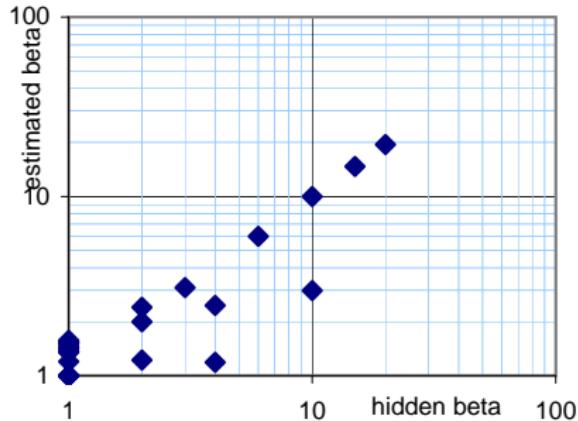
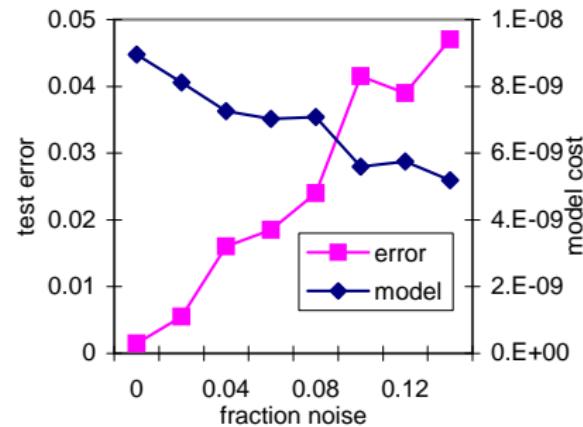
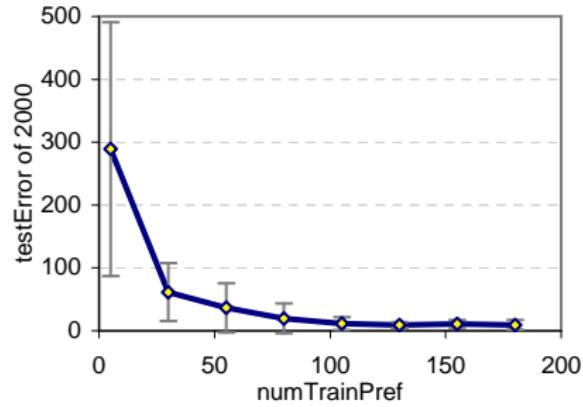
- ▶ Synthetic data
- ▶ Trainer knows and fixes all but one  $\beta_j$
- ▶ Sweep over unknown  $\beta_j$  and plot training loss
- ▶ Local minima possible, never seen
- ▶ True and smoothed global minima line up

# Convergence



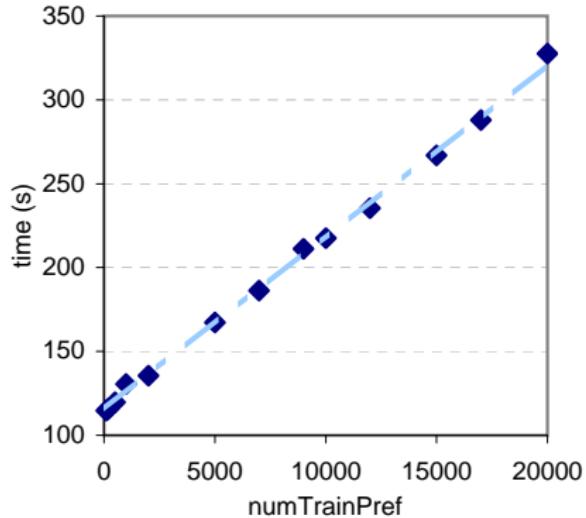
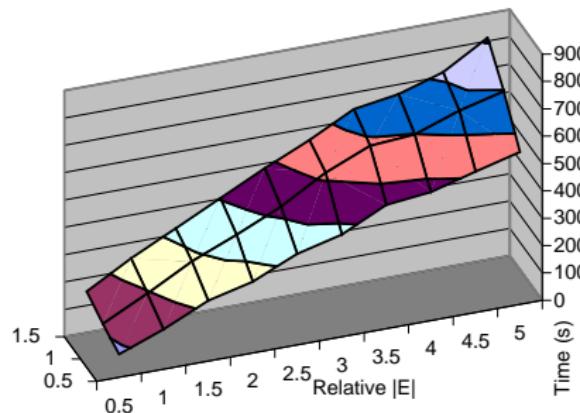
- ▶ Error in gradient decreases rapidly with PageRank-like iterations
- ▶ True and approximate losses decrease rapidly with Newton iterations

## $\beta$ estimation and learning performance



- ▶ Fast training rate
- ▶ Robust to training noise
- ▶ Reconstructs  $\beta$  reasonably

# NETRANK training performance



- ▶ Scales linearly with  $|V|$  and  $|E|$
- ▶ Scales linearly with  $|\prec|$
- ▶ Edge list and  $\prec$  can be scanned from disk

## Dynamic personalized PageRank

- ▶ PageRank  $p_r = \alpha C p_r + (1 - \alpha)r$
- ▶  $C$  learnt offline
- ▶ Sparse teleport  $r$  specified at query time
- ▶ Query goal: Quickly identify top- $k$  elements of  $p_r$ 
  - ▶ Interactive query time irrespective of  $|G|$
  - ▶ Index space can scale up as a small fraction of  $|G|$
  - ▶ Preprocessing time must be practical
- ▶ Solves to  $p_r = (1 - \alpha)(\mathbb{I} - \alpha C)^{-1} r$ , linear in  $r$ 
  - ▶  $\therefore p_{r_1+r_2} = p_{r_1} + p_{r_2}$  and  $p_{\gamma r_1} = \gamma p_{r_1}$
  - ▶ If  $r = \delta_u$  ( $r(u) = 1$ ,  $r(v) = 0 \forall v \neq u$ ),  $p_{\delta_u}$  is called PPV $_u$
  - ▶ Given PPV $_u \forall u$ , can compute  $p_r$  for any  $r$  quickly

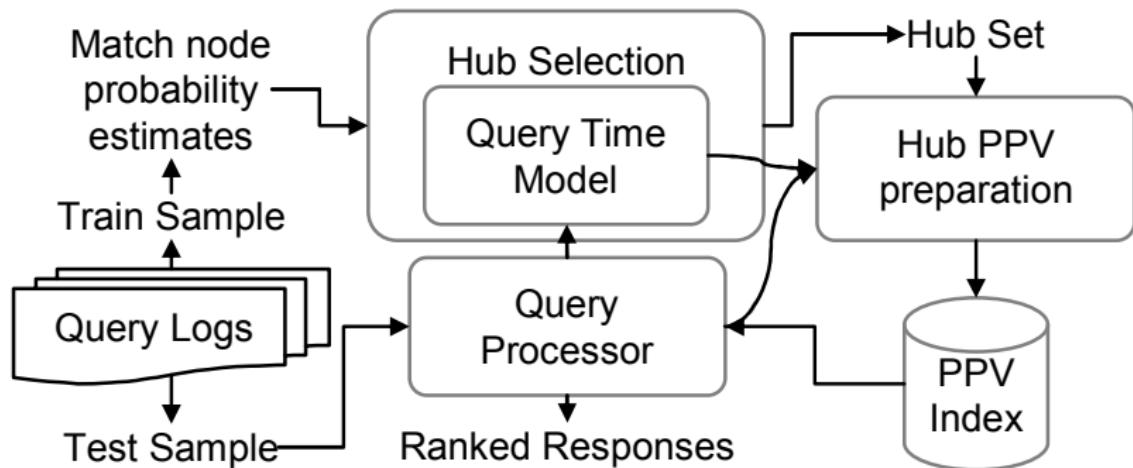
## Existing approaches vs. HUBRANK

- ▶ Query-time power iterations impractical for large graphs (too slow even for CITESEER graph)
- ▶ OBJECTRANK stores  $\text{PPV}_w$  for “all” words  $w$
- ▶ Vocabulary scales with  $V$ , impractical preprocessing time: 75k nodes, 562k words → 22,000 CPU-hours

Key idea in HUBRANK:

- ▶ Based on query workload, carefully choose nodes  $H \subset V$  for which  $\text{PPV}_h$  is precomputed and stored
- ▶ At query time, do work somewhere in between instant lookup and full-blown PageRank

# HUBRANK system diagram



# Asynchronous PageRank

$$p_r = \alpha C p_r + (1 - \alpha) r$$

$q \leftarrow r, p \leftarrow \vec{0}$   
**while**  $|q|$  is large **do**  
     $p \leftarrow p + (1 - \alpha)q$   
     $q \leftarrow \alpha C q$

$q \leftarrow r, p \leftarrow \vec{0}$   
**while** some  $q(u)$  large **do**  
     $\hat{q} \leftarrow q(u), q(u) \leftarrow 0$   
     $p(u) \leftarrow p(u) + (1 - \alpha)\hat{q}$   
**for**  $(u, v) \in E$  **do**  
         $q(v) \leftarrow q(v) + \alpha C(v, u)\hat{q}$

Advantage of asynchronous version is that it can use PPVs at hub nodes

## Stopping at hubs

$$q \leftarrow r, \hat{p} \leftarrow \vec{0}$$

**while** some  $q(u)$  is large **do**

$$\hat{q} \leftarrow q(u), q(u) \leftarrow 0$$

**if**  $u \in H$  **then**

$$\hat{p} \leftarrow \hat{p} + \hat{q} \text{PPV}_u \quad /* \hat{q} \text{ blocked */}$$

**else**

$$\hat{p}(u) \leftarrow \hat{p}(u) + (1 - \alpha)\hat{q}$$

**for**  $(u, v) \in E$  **do**

$$q(v) \leftarrow q(v) + \alpha C(v, u)\hat{q} \quad /* \hat{q} \text{ "pushed" */}$$

## Correctness

- ▶  $\hat{p} + p_q$  is invariant across the while loop
- ▶ In the beginning,  $\hat{p} = \vec{0}$ ,  $q = r$ ,  $p_q = p_r$
- ▶ At the end,  $q \approx \vec{0}$   
 $\therefore p_q \approx (1 - \alpha)(\mathbb{I} - \alpha C)^{-1}\vec{0} \approx \vec{0}$   
 $\therefore \hat{p} \approx p_r$

## Termination

- ▶ Suppose we quit when  $\|q\|_1 < \epsilon_{\text{push}}$
- ▶ Each iteration reduces  $\|q\|_1$  by  $\geq (1 - \alpha)\epsilon_{\text{push}}$
- ▶ Begin with  $\|q\|_1 = \|r\|_1$  ( $= 1$  for legal teleports)  
 $\therefore$  At most  $\|r\|_1 / (1 - \alpha)\epsilon_{\text{push}}$  loop iterations

## A more efficient implementation

```
 $q \leftarrow r, N_{H,r} \leftarrow \vec{0}, B_{H,r} \leftarrow \vec{0}$ 
while  $\|q\|_1 > \epsilon_{\text{push}}$  do
    pick node  $u$  with largest  $q(u) > 0$           /* delete-max */
     $\hat{q} \leftarrow q(u), q(u) \leftarrow 0$ 
    if  $u \in H$  then
         $B_{H,r}(u) \leftarrow B_{H,r}(u) + \hat{q}$           /* blocker score */
    else
         $N_{H,r}(u) \leftarrow N_{H,r}(u) + (1 - \alpha)\hat{q}$ 
        for each out-neighbor  $v$  of  $u$  do
             $q(v) \leftarrow q(v) + \alpha C(v, u)\hat{q}$           /* increase-key */
    return  $N_{H,r} + \sum_{h \in H} B_{H,r}(h) \text{PPV}_h$ 
```

$N_{H,r}(u)$  is PageRank transmitted from  $r$  to  $u$  bypassing  $H$

## Cost-benefit consideration

If  $u$  is included in  $H$ ,

- ▶ Will need space to store  $\text{PPV}_u$
- ▶ Will save pushes

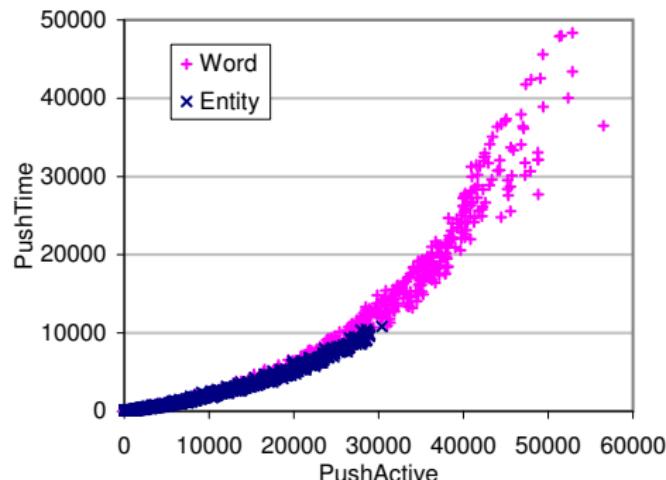
How much? Depends on ...

- ▶ How often residuals  $q$  reach from queries to  $u$
- ▶ Which depends on what other nodes are in  $H$
- ▶ Time needed to push weights out from  $u$
- ▶ What other nodes are already in  $H$

Estimate  $\text{PushTime}(H, \delta_u, \epsilon_{\text{push}})$  without pushing!

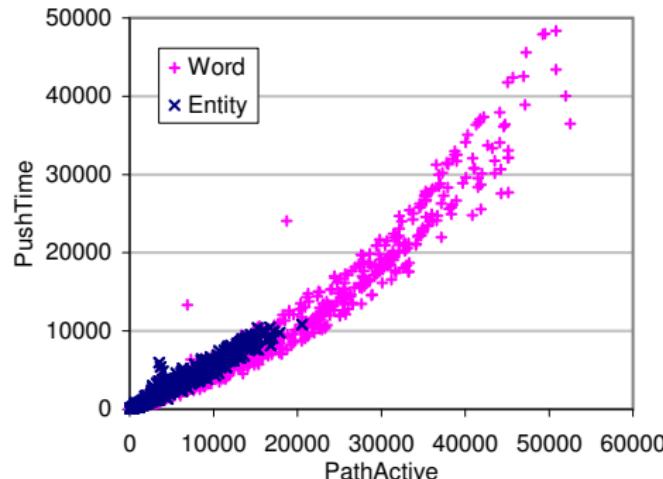
New models and algorithms to estimate query time benefits and space cost

## $\text{PushActive}(H, \delta_u, \epsilon_{\text{push}})$



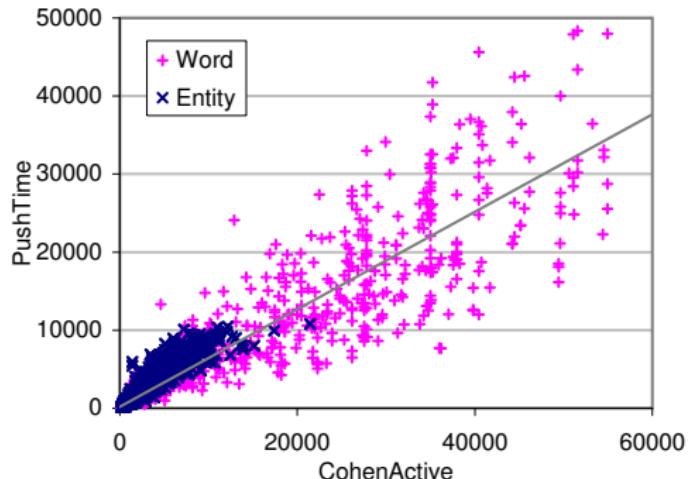
- ▶  $\text{PushActive}(H, \delta_u, \epsilon_{\text{push}})$  is the number of distinct nodes from which residual was pushed out
- ▶ Predicts PushTime( $H, \delta_u, \epsilon_{\text{push}}$ ) remarkably well
- (⌚)  $\text{PushActive}(H, \delta_u, \epsilon_{\text{push}})$  known only after pushing

## PathActive( $H, \delta_u, \epsilon_{\text{push}}$ )



- ▶ Number of nodes reachable from  $u$ , avoiding  $H$ , on a path having conductance at least  $\epsilon_{\text{push}}$
- ▶ Good predictor of  $\text{PushTime}(H, \delta_u, \epsilon_{\text{push}})$
- ▶  $O(|V|^2 \log |V| + |V| |E|)$  to compile  $\forall u$ , slow

## CohenActive( $H, \delta_u, \epsilon_{\text{push}}$ )



- ▶ Monte Carlo estimator for PathActive( $H, \delta_u, \epsilon_{\text{push}}$ )
- ▶ Takes expected  $O(|E| \log |V| + |V| \log^2 |V|)$  time
- ▶ Accuracy acceptable for index optimization
- ▶ Output: **WorkSaved( $H, \delta_w, u$ )** — work saved while finding  $p_{\delta_w}$  because  $u$  is included in  $H$

## From $\text{WorkSaved}(H, \delta_u, u)$ to workload

- ▶ Linear approx acceptable:  $\text{PushTime}(H, r, \epsilon_{\text{push}})$

$$\approx \sum_u \text{PushTime}(H, r(u)\delta_u, \epsilon_{\text{push}})$$

- ▶ Let  $\tilde{f}(w)$  be the probability that there is a teleport to node  $w$

$$\text{WorkSaved}(H, u) \approx \sum_w \tilde{f}(w) \text{WorkSaved}(H, \delta_w, u)$$

- ▶ Final expression for worth of including  $u$  in  $H$  averaged over workload:

$$\text{PushTime}(H, \delta_u, \epsilon_{\text{push}}) \sum_w \tilde{f}(w) N_{H, \delta_w}(u)$$

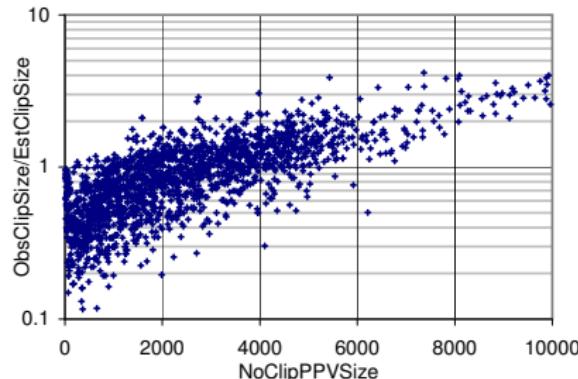
## From $\text{WorkSaved}(H, \delta_u, u)$ to workload (2)

- Advantage: can exploit linearity and write

$$\begin{aligned} & \text{PushTime}(H, \delta_u, \epsilon_{\text{push}}) \sum_w \tilde{f}(w) N_{H, \delta_w}(u) \\ & \approx N_{H, \tilde{f}}(u) \text{PushTime}(H, \delta_u, \epsilon_{\text{push}}) \end{aligned}$$

- How often pushes from query nodes reach  $u$
- Push time needed “downstream” from  $u$

## Space cost of including $u$ in $H$

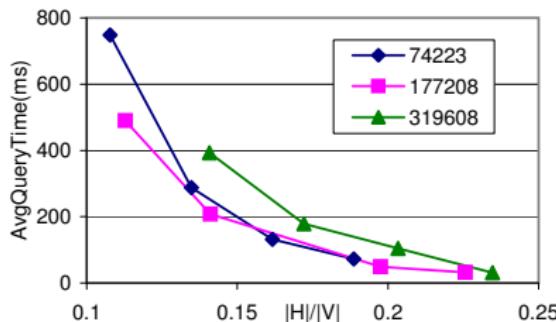


- ▶ Unclipped PPV may be  $|V|$  doubles, too big
- ▶ Clip any element less than  $\epsilon_{clip}$ : much smaller
- ▶ Challenge: How does optimizer know size of PPV without computing it?
- ▶ Exploit property that PPV elements are distributed according to power law

## Cost-benefit optimizer

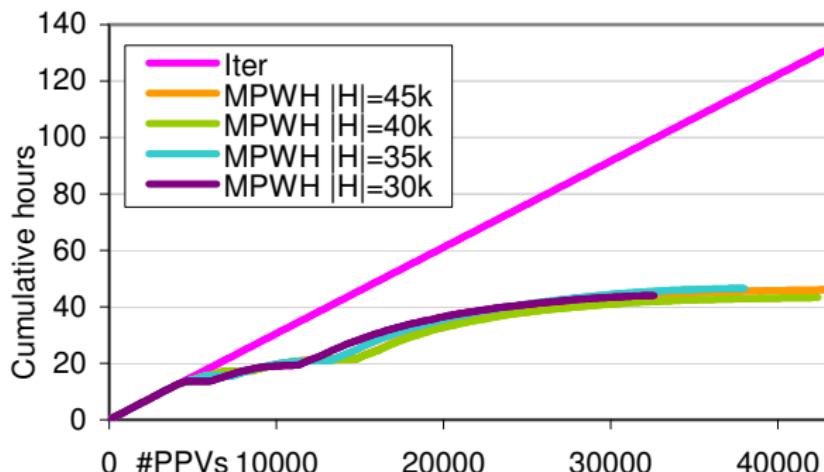
**Inputs:** target  $|H|$ , PPV clip threshold  $\epsilon_{\text{clip}}$ , batchSize  
find CohenActive( $\emptyset, \delta_u, \epsilon_{\text{push}} = 0$ ) for all  $u \in V$   
thus estimate clipped sizes of all  $\text{PPV}_u$  /\* cost \*/  
 $H \leftarrow \emptyset$   
**while**  $H$  is not large enough **do**  
    for all nodes  $u$  not in  $H$  yet compute /\* benefit \*/  
     $\sum_w \tilde{f}(w) \text{WorkSaved}(H, \delta_w, u)$   
    set  $\text{merit}(u) = \text{benefit}(u)/\text{cost}(u)$   
    greedily include best batchSize nodes in  $H$

## HUBRANK query performance & scaling



- ▶ Lucene text index takes **55 MB**
- ▶ OBJECTRANK indexing takes **22,000 hours**
- ▶ OBJECTRANK typical query time **8–10 seconds**
- ▶ HUBRANK index takes **5–15 MB**
- ▶ HUBRANK indexing takes  $\sim 20$  hours
- ▶ HUBRANK query time is **12–200 ms**
- ▶ Scaling  $G$  has little effect if  $|H|/|V|$  is held fixed

## Preparing the PPV index



- ▶ Carefully order  $h \in H$
- ▶ For each  $h$  choose Power Iterations vs. BCA
- ▶ Earlier PPVs speed up computing later PPVs
- ▶ MPWH: Maximum personalized PageRank wrt  $H$
- ▶ Total indexing time levels off as  $H$  grows

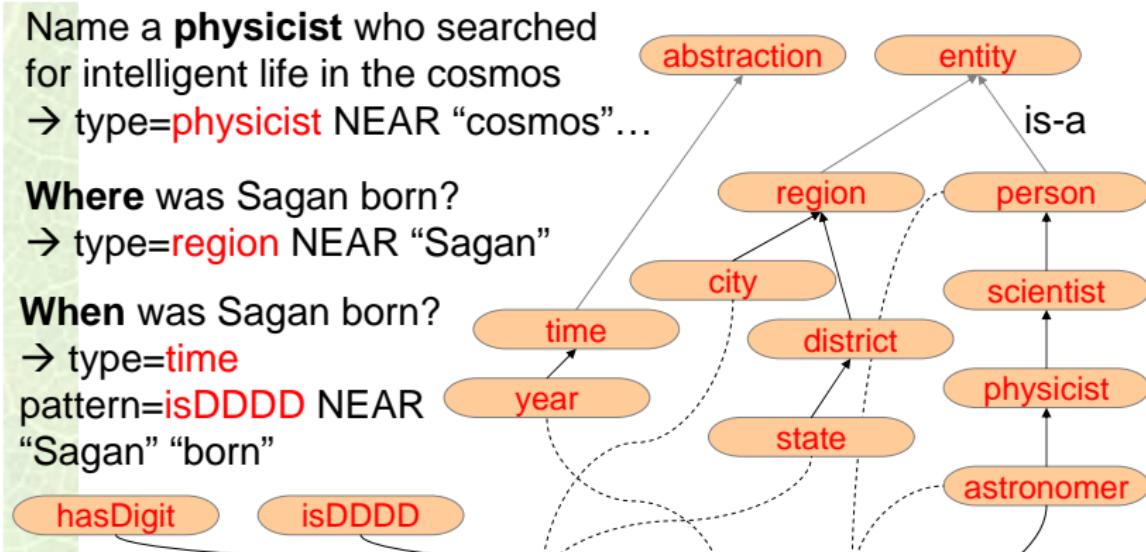
# Querying text with type annotations

Name a **physicist** who searched for intelligent life in the cosmos  
→ type=**physicist** NEAR "cosmos"...

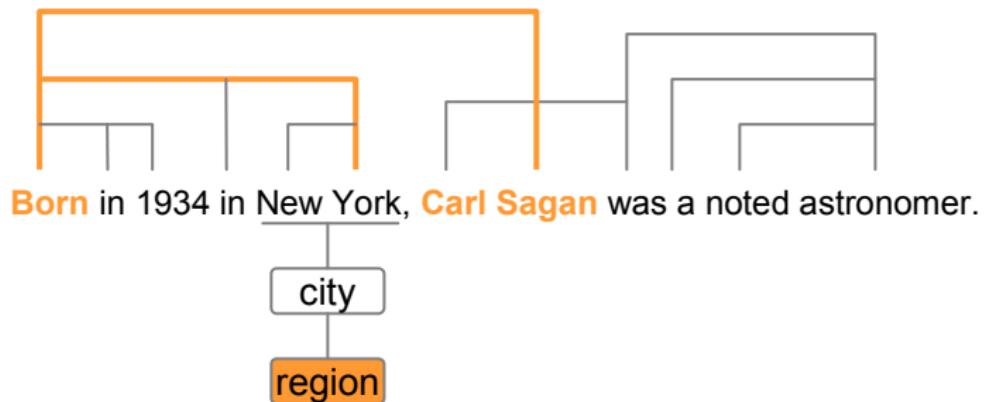
**Where** was Sagan born?  
→ type=**region** NEAR "Sagan"

**When** was Sagan born?  
→ type=**time**  
pattern=**isDDDD** NEAR "Sagan" "born"

Born in New York in 1934, Sagan was a noted astronomer whose lifelong passion was searching for intelligent life in the cosmos.



## Graphs from parsers and type taxonomies



- ▶ Dependency edges are typed (not shown)
- ▶ As before, query matches some words and types
- ▶ Proximity is now non-linear, via dependency edges

## System design dimensions

- ▶ Use of catalogs, types, entities
  - ▶ How many types? entities?
  - ▶ Are entities and types canonicalized or kept (open) strings?
  - ▶ Disjoint or hierarchical?
  - ▶ Annotation perfect or probabilistic?
- ▶ Query language
  - ▶ Un/ordered windows, phrases, other evidence patterns
  - ▶ Variables, bindings
  - ▶ Variable and type associations
  - ▶ Select clauses, joins
  - ▶ Implicit or explicit aggregate operators

## System design dimensions (2)

- ▶ Indexing and query processing
  - ▶ Posting lists for entity mentions
  - ▶ Posting lists for types?
  - ▶ Managing redundancy and replication
  - ▶ Proximity-cognizant indices
  - ▶ Distributed processing issues
- ▶ Entity/relation tuple ranking
  - ▶ Window/snippet scoring semantics
  - ▶ Semantics of aggregation over snippets
  - ▶ Semantics of aggregation over subqueries

## Single entity queries

In IR4QA (2004)

- ▶ Motivated by factoid queries in question answering
- ▶ “Who was the first man in space?”
- ▶ Type = person (or astronaut?)
- ▶ Selectors = first, space
- ▶ Some magic window width

Possible embellishments:

- ▶ Allow known entities
- ▶ “Where was Einstein born?”
- ▶ Allow entities and types in phrases
- ▶ “Einstein went to type=place”

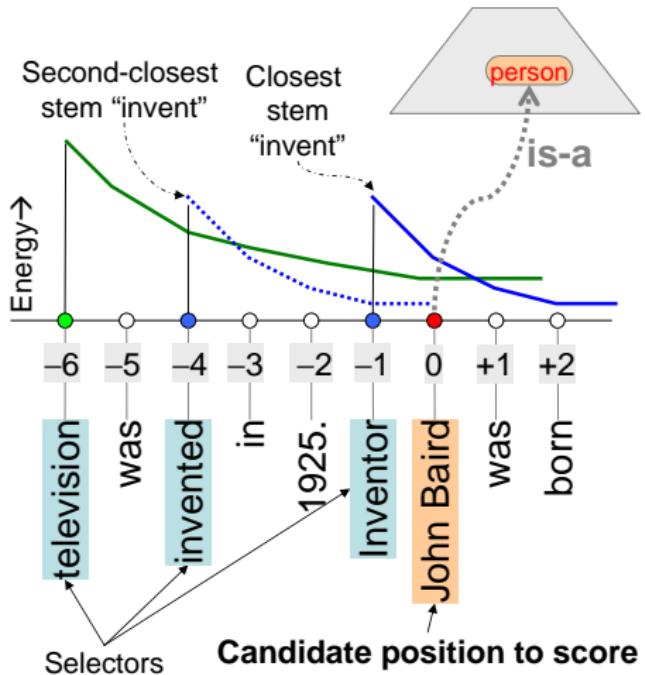
## CSAW query language

- ▶ Influenced by IR4QA, ENTITYRANK, NAGA
- ▶ Basic components: variables, entities, types, literals
- ▶ Types can be quantities, important special case
- ▶ Constraints: contexts, equijoins
- ▶ Example:  $m \in^+ \text{Movie}$ ,  $p \in^+ \text{MoneyAmount}$ ,  
 $a \in^+ \text{CountedNumber}$
- ▶  $\text{uw}(m, p, \text{budget}, \text{production})$
- ▶  $\text{uw}(m, a, \text{academy}, \text{oscar}, \text{award})$
- ▶ Note that *different* contexts may satisfy the subqueries
- ▶ Shared variable  $m$  implements an equijoin
- ▶ Ranking semantics — not yet decided

# Learning proximity scores of token spans

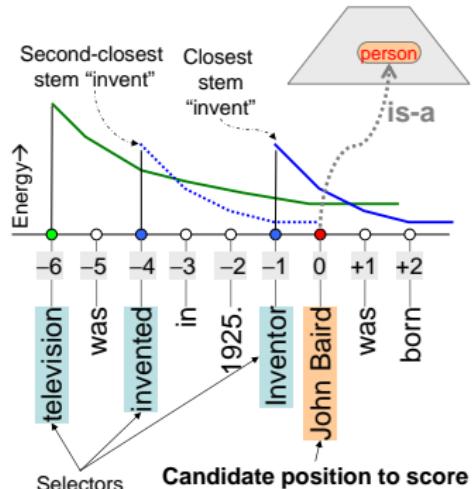
```
type=person NEAR "television" "invent*"
```

- ▶ Rarity of selectors
- ▶ Distance from candidate position to selectors
- ▶ Many occurrences of one selector (closest)
- ▶ Combining scores from many selectors (sum)



## Feature vector $x$ for position 0

- ▶ Limit to  $\pm W$  window
- ▶  $x(-4) = 0; x(-1) = \text{IDF}(\text{invent}^*); x(-6) = \text{IDF}(\text{television})$
- ▶  $\text{IDF}(w) = \text{numDocs}/\text{numDocsWith}(w)$ , or perhaps  $\text{IDF}(w) = \log(1 + \text{numDocs}/\text{numDocsWith}(w))$
- ▶ Other features, e.g., is selector noun? candidate has digits?  
(If in doubt, throw everything into the kitchen sink)



## Training from relevance judgments

- ▶ For each query  $q$ , there is a set of items (documents, snippets, entities, ...)  $D_q$
- ▶  $D_q^+ \subset D_q$  is good (relevant),  $D_q^- = D_q \setminus D_q^+$  is bad (irrelevant)
- ▶  $n_q^+ = |D_q^+|, n_q^- = |D_q^-|$
- ▶ Can use TREC QA data for entity search
- ▶ Each good item  $g$  preferred to each bad item  $b$ :

$$g \succ b \quad \text{or equivalently} \quad b \prec g$$

## Training from relevance judgments

- ▶ Item  $i$  characterized by feature vector  $x_{qi} \in \mathbb{R}^d$  that depends on query
- ▶ Learner estimates **model**  $\beta \in \mathbb{R}^d$
- ▶ Given a test query, good/bad not known
- ▶ **Score** of item is dot product  $f_{\beta}(x_{qi}) = \beta^\top x_{qi}$
- ▶ Sort items by decreasing score, present top- $k$

## Learning to rank

- ▶ Evaluation criteria: pair preference violation, area under curve (AUC), mean average precision (MAP)
- ▶ Not the same as two-class classification
- ▶ RANKSVM: learning  $\beta$  to minimize pair preference violation
- ▶ Learning  $\beta$  to maximize AUC
- ▶ (Can also directly maximize MAP)

## The model vector $\beta$

- ▶ Score of candidate position is  $\beta x$
- ▶  $\beta(j)$  is the value of the decay function at offset  $j$
- ▶ TREC gives us correct  $i$  and incorrect  $j$  token spans
- ▶  $i \prec j$  means we want  $\beta x_i + \text{margin} \leq \beta x_j$
- ▶ Want  $\beta$  to be smooth, with  $\beta(-W - 1) = \beta(W + 1) = 0$

$$\min_{\beta} \sum_{j=-W}^{W+1} (\beta_{j-1} - \beta_j)^2 + B \sum_{i \prec j} \text{SmoothLoss}(\beta x_i + 1 - \beta x_j)$$

$$\min_{\beta} \sum_{j=-W}^{W+1} (\beta_{j-1} - \beta_j)^2 + B \sum_{i \prec j} \log(1 + \exp(\beta x_i + 1 - \beta x_j))$$

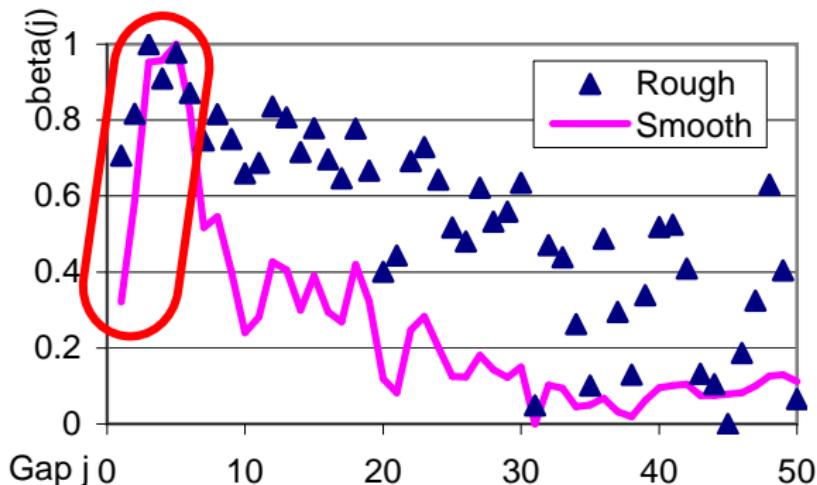
## The model vector $\beta$

- ▶ Score of candidate position is  $\beta x$
- ▶  $\beta(j)$  is the value of the decay function at offset  $j$
- ▶ TREC gives us correct  $i$  and incorrect  $j$  token spans
- ▶  $i \prec j$  means we want  $\beta x_i + \text{margin} \leq \beta x_j$
- ▶ Want  $\beta$  to be smooth, with  $\beta(-W - 1) = \beta(W + 1) = 0$

$$\min_{\beta} \sum_{j=-W}^{W+1} (\beta_{j-1} - \beta_j)^2 + B \sum_{i \prec j} \text{SmoothLoss}(\beta x_i + 1 - \beta x_j)$$

$$\min_{\beta} \sum_{j=-W}^{W+1} (\beta_{j-1} - \beta_j)^2 + B \sum_{i \prec j} \log(1 + \exp(\beta x_i + 1 - \beta x_j))$$

## $\beta$ fit to TREC QA



Train	Test	MRR
IR	2000	0.16
2001	2000	0.29

'2001'='TREC 2001';  
'MRR'=Mean reciprocal rank

- ▶ Only positive offsets shown
- ▶ Unexpected decay shape!
- ▶ Smooth function better than rough function
- ▶ Both better than 3-sentence TFIDF

## Features for snippet scoring

- ▶ Not common to fit proximity function as above
- ▶ Instead approximate by simple parameterized (usually monotone decreasing) decay function
- ▶ Throw together a bunch of “positive evidence” features
  - ▶ TFIDF cosine match between snippet and query (selectors only)
  - ▶ (Weighted) Jaccard in place of TFIDF cosine
  - ▶ Positional language models (Zhai)
- ▶ An important “**negative evidence**” feature
  - ▶ Suppose you are looking for scientists who played the violin
  - ▶ If a snippet mentions played, violin and only Einstein, it is good evidence
  - ▶ But if a snippet mentions Einstein, Bohr and Rutherford, it gives less evidence for each

## Other entity scoring models: Proximity kernels

- ▶ In standard PIR,  $\Pr(t|d) = \frac{1}{N} \sum_{i=1}^N \delta_d(i, t)$  where  $N$  is the length of the document  $d$ ,  $t$  is a term and  $\delta_d : \mathbb{N} \times T \rightarrow \{0, 1\}$  such that

$$\delta_d(i, t) = \begin{cases} 1, & \text{if term at position } i \text{ in document } d \text{ is } t, \\ 0, & \text{otherwise} \end{cases}$$

- ▶ Extend to a kernel function  $k(i, c)$  that decays with  $|i - c|$ :

$$\Pr_k(t|c, d) = \frac{\sum_{i=1}^N \delta_d(i, t) k(i, c)}{\sum_{i=1}^N k(i, c)}$$

where  $c$  is the position where a candidate entity appears in the document

- ▶ Language model is biased toward words near candidate entity
- ▶ Earlier we trained  $k(i, c)$  from preferences

## Other entity scoring models: Proximity kernels (2)

- ▶ The bigger issue is assumptions of term (in)dependence in supporting the candidate entity
- ▶ If the set of query terms is  $T$  and query terms  $t_i \in T$  are independent given a candidate  $c$ , then

$$\Pr(T|c) \propto \prod_{t_i \in T} \left\{ \sum_{d \in D} \Pr(d|c) \Pr(t_i|c, d) \right\}$$

i.e., straight marginalization over documents  $d \in D$

- ▶ Different documents can generate different query terms
- ▶ Consider  $T = \{\text{ontology}, \text{engineering}\}$  and these two snippets

## Other entity scoring models: Proximity kernels (3)

.....

Thanks in advance.

=====

XXXXXXX X XXXXX, Ph.D.

Department of Electrical **Engineering**

XXXXXX University

Slide list: The Semantic Web

by XXXX XXXXXX

Slide: Digital Libraries and the Semantic Web

Formalized facilities for supporting **ontology**

merging .....

- ▶ Alternatively, we can assume terms are independent given a document, then we get

$$\Pr(T|c) \propto \sum_{d \in D} \left\{ \prod_{t_i \in T} \Pr(t_i|c, d) \right\} \Pr(d|c)$$

## Other entity scoring models: Proximity kernels (4)

- ▶ Each document has to plausibly generate all query terms
- ▶ Might be too harsh and lose recall
- ▶ Perhaps best to blend the two models?
- ▶ But a fixed blend irrespective of query (subclauses) may not be best
- ▶ Depends on finer structure of the query
  - ▶ “Indian politician” — evidence for this may come from different documents, or entity name may reveal “Indian”
  - ▶ ... whose oil paintings sell for actual money: here oil and painting must match in a single document

## Two-hop relevance

- ▶ Entities  $e$  may be mentioned in documents  $d$
- ▶ Let the salience of  $e$  in  $d$  be a binary random variable  
 $R_{de} \in \{0, 1\}$
- ▶ Document  $d$  may be relevant to query  $q$
- ▶ Let the relevance of  $d$  to  $q$  be a binary random variable  
 $R_{qd} \in \{0, 1\}$  (as in standard IR)
- ▶ Define the score of  $e$  given  $q$  as

$$\sum_d \Pr(d) \Pr(R_{qd} = 1 | q, d) \Pr(R_{de} = 1 | d, e)$$

- ▶  $\Pr(d)$  is a prior or static score
- ▶ Note that in training data we just have  $q, \{e_+\}, \{e_-\}$
- ▶ Viz., queries, pos ents, neg ents
- ▶ Too laborious to label relevance of snippets or documents

## Two-hop relevance (2)

- ▶ Model the two relevance distributions using logistic regression:

$$\Pr(R_{qd} = 1 | q, d) = \sigma(\alpha \cdot f(q, d))$$

$$\Pr(R_{de} = 1 | d, e) = \sigma(\beta \cdot g(d, e)), \quad \text{where}$$

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

- ▶  $f, g$  are feature vectors
- ▶ In particular,  $g$  is similar to entity disambiguation features
- ▶  $f$  follows standard probabilistic IR feature design
- ▶  $\alpha, \beta$  are corresponding trained model parameters
- ▶ Optimization not convex

## Design of $f(q, d)$

- ▶ BM25 match score between  $q$  and  $d$
- ▶ Other IR scores like TFIDF cosine, language model
- ▶ Match between  $q$  and anchor text pointing to  $d$
- ▶ Match between  $q$  and title of  $d$
- ▶ PageRank of  $d$  (independent of  $q$ )
- ▶ Length of URL of  $d$  (independent of  $q$ )
- ▶ Note,  $f(q, d) \geq \vec{0}$
- ▶ If static scores were removed, and there was absolutely no match between  $q$  and  $d$  in any form,  $f(q, d) = \vec{0}$
- ▶ Note that  $\sigma(\alpha \cdot \vec{0}) = 1/2$
- ▶ Therefore we add a constant feature and let the corresponding  $\alpha_j$  set up a (negative) bias

## Design of $g(d, e)$

(In the context of person mentions)

- ▶ Exact full name of  $e$  appears in  $d$
- ▶ Last name of  $e$  appears in  $d$
- ▶ Email of  $e$  (if you know that) appears in  $d$
- ▶ (Above are specific to person entities in electronic media)
- ▶ Salience of candidate  $e$  in document  $d$
- ▶ A great deal differs depending on whether
  - ▶ the candidate entity is expected to be in the KG, and has been annotated in the corpus snippets, or
  - ▶ the candidate entity is described by variant strings/spans in different snippets (the OpenIE approach) and these need to be reconciled at query time

## Joint answer ranking

- ▶ For candidate entities that are not canonical entity IDs
- ▶ Fine-type tag a corpus
- ▶ Infer target type from query
- ▶ Use a positional language model to collect snippets
- ▶ Then collect candidate entity mention strings  $\{x_i\}$
- ▶ How to rank these?
- ▶ Independent prediction  $\Pr(y_i = 1|q, x_i)$
- ▶ Joint prediction  $\Pr(y_i = 1|q, \{x_i : i = 1, \dots, n\})$
- ▶ Model as

$$\Pr(y_1, \dots, y_n | \{x_i\}) \propto \sum_{i=1}^n \left[ \beta \cdot \phi(x_i, y_i) + \sum_{j \in N(i)} \lambda \cdot \psi(x_i, x_j; y_i, y_j) \right]$$

## Joint answer ranking (2)

- ▶ Vector  $\phi$  contains features measuring compatibility between  $x_i$  and  $y_i$ ; could be a standard logistic model
- ▶ Vector  $\psi$  contains features measuring compatibility between  $x_i$  and  $x_i$ , given proposed labels  $y_i, y_j$ 
  - ▶ If  $y_i = y_j = 1$ , compatibility is large if  $x_i \approx x_j$
  - ▶ If  $y_i = y_j = 0$ , cannot really say  $x_i$  and  $x_j$  must be dissimilar (so ignore)
  - ▶ If  $y_i \neq y_j$ , might say very different  $x_i, x_j$  are more compatible (but sometimes this case is ignored too)
- ▶ Results in a Boltzmann machine, for which good  $\beta^*, \lambda^*$  can be learnt

## Joint answer ranking: inference

- ▶ By pinning respectively to 0 and 1, we can also find marginal probabilities for each  $x_i$ , i.e.  $\Pr(y_i|\{x_i\}; \beta^*, \lambda^*)$
- ▶ A problem with ranking by marginals is that variant mentions in a cluster may be listed one after another because they have marginal scores close together, e.g., Bill Clinton, William J. Clinton, Clinton, WJC
- ▶ Suppose we find marginals of remaining candidates  $x_j$  assuming results reported thus far all have  $y_i = 1$
- ▶ And we also compute unconditioned marginal of  $y_j$
- ▶ For strongly similar answers, these two marginals are likely to be close to each other
- ▶ Unconditional minus conditioned gives measure of novelty

## Joint answer ranking: inference (2)

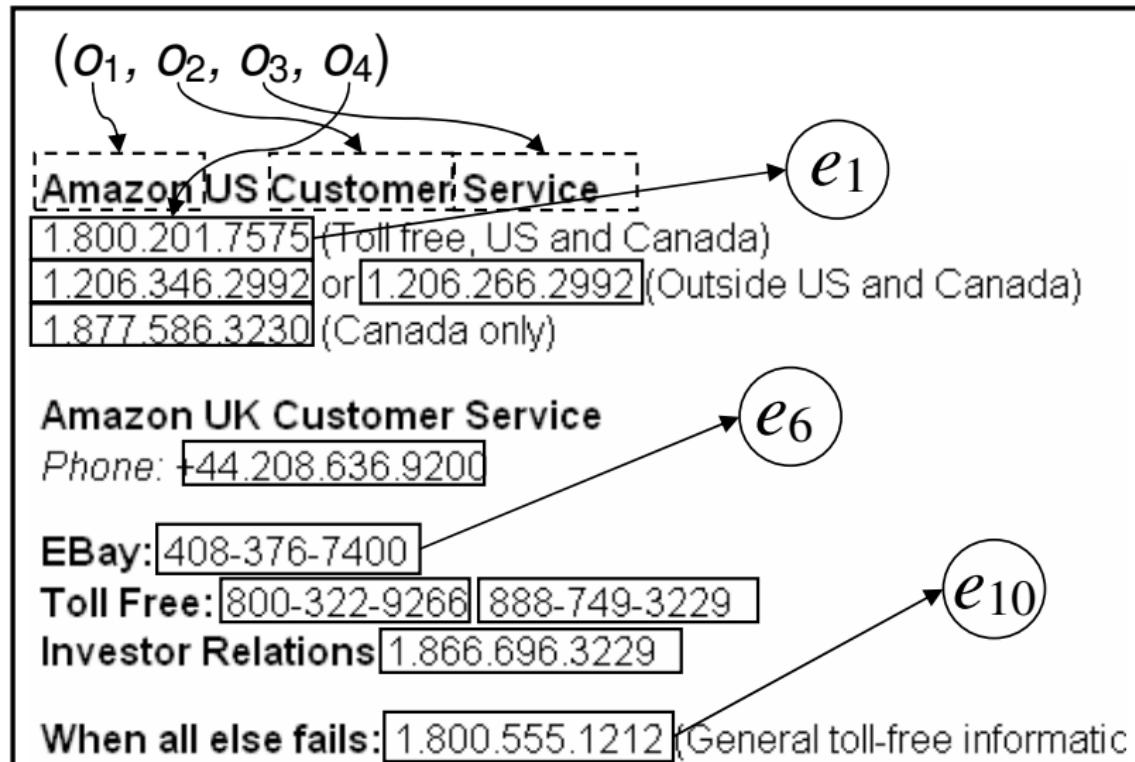
- ▶ This inspires the ranking pseudocode

```
1: create empty answer pool  
2: for each answer  $x_i$  do  
3:   estimate  $\Pr(y_i = 1 | \mathbf{x})$   
4: move max score answer to pool  
5: while pool not big enough do  
6:   for remaining candidates  $x_j$  do  
7:     calculate conditional  $\Pr(y_j = 1 | \mathbf{x}; \text{pool is all correct})$   
8:     calculate the lift of  $x_j$  as  
            $\Pr(y_j = 1 | \mathbf{x}) - (\text{above conditional})$   
9:   move  $x_j$  with largest lift into pool
```

## ENTITY RANK

- ▶ Ad-hoc record extraction from single snippet
- ▶ (Plus, evidence aggregation over snippets)
- ▶ “Find institutions and email addresses of database researchers”
- ▶ `doc(o1, ..., om)`: objects o<sub>i</sub> must all occur in the same document
- ▶ `phrase(o1, ..., om)`: objects o<sub>i</sub> must all occur in exactly the same sequence (i.e., order and adjacency) as specified
- ▶ `uw(n)(o1, ..., om)`: objects o<sub>i</sub> must all occur in a window of no more than n words (default: document length)
- ▶ `ow(n)(o1, ..., om)`: in addition to uw, o<sub>i</sub> must all occur in the given order

## ENTITYRANK example source page



Query: Amazon Customer Service #phone

## ENTITYRANK ranking considerations

- ▶ Ranking of source page/s expressed as “prior” probability  $\Pr(d)$
- ▶ Could be related to PageRank
- ▶ Or rank as per page-level search engine, as in  $\propto \blacksquare^{-\text{rank}}$
- ▶ A snippet(/page) level score for an entity, written  $\Pr(e|d, q)$ , which depends on these signals
  - ▶ Entity extraction confidence
  - ▶ Lexical proximity between candidate entity and query matches
  - ▶ (Background rate of occurrence of entity  $e$ )
- ▶ Roughly,  $\Pr(e|q) = \sum_d \Pr(d) \Pr(e|q, d)$
- ▶ Let the query have types  $T_1, \dots, T_m$  (not necessarily distinct)
- ▶ E.g.  $T_1 = \text{scientist}, T_2 = \text{musical instrument}$
- ▶  $\gamma_d$  is an assignment of entities  $e_j \in T_j$  such that some snippet in  $d$  supports it

## ENTITY RANK ranking considerations (2)

- ▶ Each entity mention found in  $\gamma$  has a **confidence** of extraction called  $\text{conf}(e_j, \gamma)$
- ▶ Take the *best* assignment from each doc (how?)
- ▶  $p_o = \Pr(e|q) = \sum_d \Pr(d) \max_{\gamma_d} \prod_j \text{conf}(e_j, \gamma_d)$
- ▶ An interesting innovation is the use of a baseline probability  $p_r$  of finding entity  $e_j$  “at random” ▶ HW
- ▶ Let  $p_o, p_r$  be observed and baseline probabilities
- ▶ Final score is  $p_o \log \frac{p_o}{p_r} + (1 - p_o) \log \frac{1 - p_o}{1 - p_r}$

## Evidence aggregation: summary

- ▶ Same entity or entity tuple endorsed by multiple snippets
  - ▶ Einstein may appear close to played, violin 5M times
  - ▶ With low snippet scores
  - ▶ Whereas Rutherford may appear close to played, violin 1M times
  - ▶ But with larger average scores
  - ▶ Background rate of occurrence of Einstein and Rutherford may be 50M and 2M
- ▶ How to combine these evidences?
  - ▶ IR4QA did trivial sum aggregation
  - ▶ Nyberg *et al.* used mincut approach based on graphical model
  - ▶ Qin *et al.* used Laplacian smoothing
  - ▶ ENTITYRANK aggregated over pages weighted by PageRank
  - ▶ Expert search used arithmetic and geometric mean after multiplying probabilities from two logistic regressions
  - ▶ SSQ used soft-OR
  - ▶ NAGA generalized language model IR

## Indexing for KG+corpus search

## Indexing for is-a + proximity search

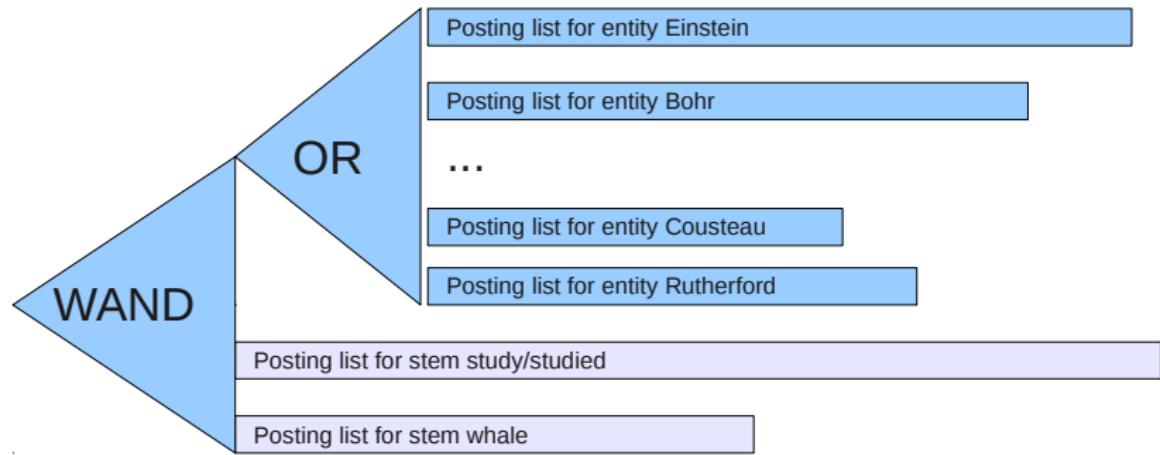
“Which scientist studied whales?” →

type=scientist NEAR study|studied whale\*

- ▶ Expand **scientist** to all instances at query time
  - ▶ Runtime type expansion too expensive
  - ▶ Even WordNet knows 650 scientist, 860 cities, ...
- ▶ Insert a pseudo-token for each hypernym of each mention of **scientist** at the same token offset as the mention
  - ▶ Problem: Index blows up
  - ▶ Open-domain type hierarchies very large: 15000 internal and 80000 leaf types in WordNet (full set  $A$ )

Solution: Index a carefully chosen small subset  $R \subset A$ , and do a little more work at query time

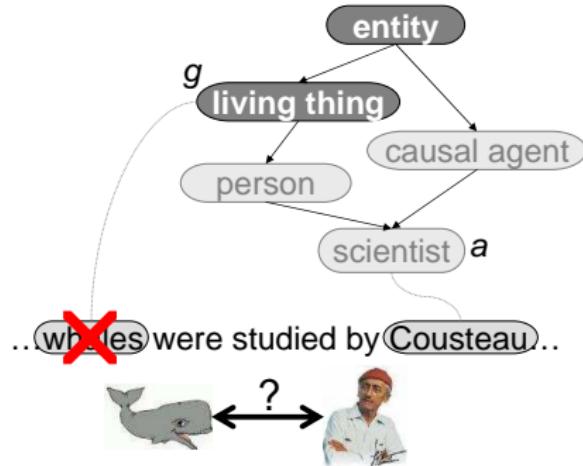
## Basic tradeoff



- ▶ Rescanning stems for each entity is impractical
- ▶ Dynamic OR of entity lists takes  $O(\log \text{fan-in})$  time
- ▶ Dynamically output OR posting has no skip info  
∴ merging with stem lists slower than usual
- ▶ Better to **materialize** often-required OR lists
- ▶ Note, much more expensive than materializing *intersections* like *New and York* where results are **smaller**

## Pre-generalize

- ▶ Index a subset  $R \subset A$
- ▶ Query atype  $a \notin R$ , want  $k$  answers
- ▶ Probe index with  $g$ , ask for  $k' > k$



## Post-filter

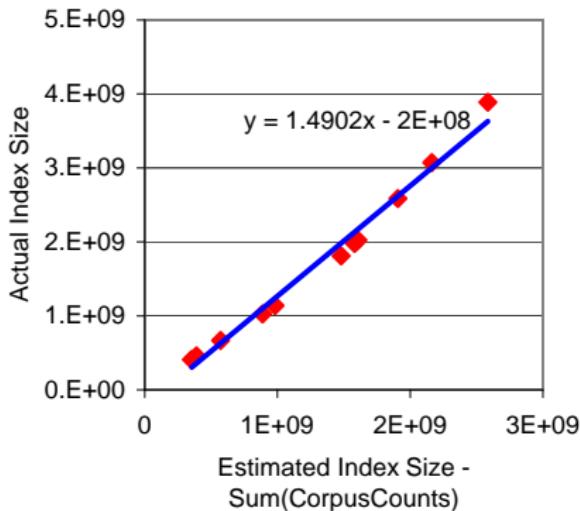
- ▶ Fetch  $k'$  high-scoring spans  $w$  for  $g$
- ▶ Check if  $w$  is-a  $a$  as well (using forward and reachability index); if not, discard
- ▶ If fewer than  $k$  survive, restart with larger  $k'$  (expensive!)

## Cost-benefit considerations

- ▶ How much space saved by indexing  $R$  instead of the whole of  $A$ ?
  - ▶ Cannot afford to try out many  $R$ s, need quick estimate
- ▶ What is the average query time bloat owing to  $a \rightarrow g$  pre-generalize and post-filter?
  - ▶ Cannot afford to test and filter too many preliminary responses

## Index space estimate

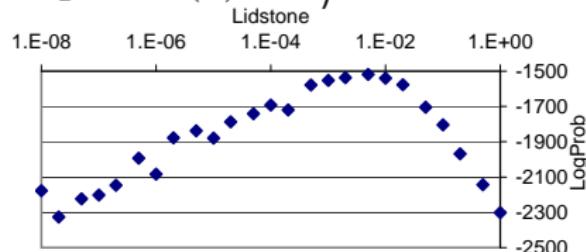
- ▶ Let  $corpusCount(a)$  be the count of tokens  $w$  in the corpus such that  $w$  is-a  $a$
- ▶ One posting entry for each count of each type  $a$
- ▶ Space estimate is  $\propto \sum_{a \in R} corpusCount(a)$
- ▶ Surprisingly accurate despite index compression



## Characterizing a query workload

$$\widetilde{\Pr}(a) = \frac{\text{queryLogCount}(a) + \lambda}{\sum_{a' \in A} (\text{queryLogCount}(a') + \lambda)}$$

- ▶ Heavy-tailed type distribution in queries
- ▶ Many test types never seen in training types and vice versa
- ▶  $\lambda = 0$  would give these types zero probability
- ▶ Danger of allocating no  $g$  close to these  $a$ s
- ▶ Build multinomial model over  $a$  with positive  $\lambda$
- ▶ Cross-validate likelihood of held-out test log



## Query time bloat estimate

- ▶  $t_{\text{scan}}$  time to scan one candidate position while merging postings
- ▶  $t_{\text{filter}}$  time to check if  $w$  is-a  $a$
- ▶ If  $R = A$  (all types indexed), query takes time roughly  
 $t_{\text{scan}} \text{corpusCount}(a)$
- ▶ If  $a \notin R$ , the price paid for generalization to  $g$  consists of
  - ▶ Longer scans:  $t_{\text{scan}} \text{corpusCount}(g)$
  - ▶ Post-filtering  $k'$  responses:  $k' t_{\text{filter}}$

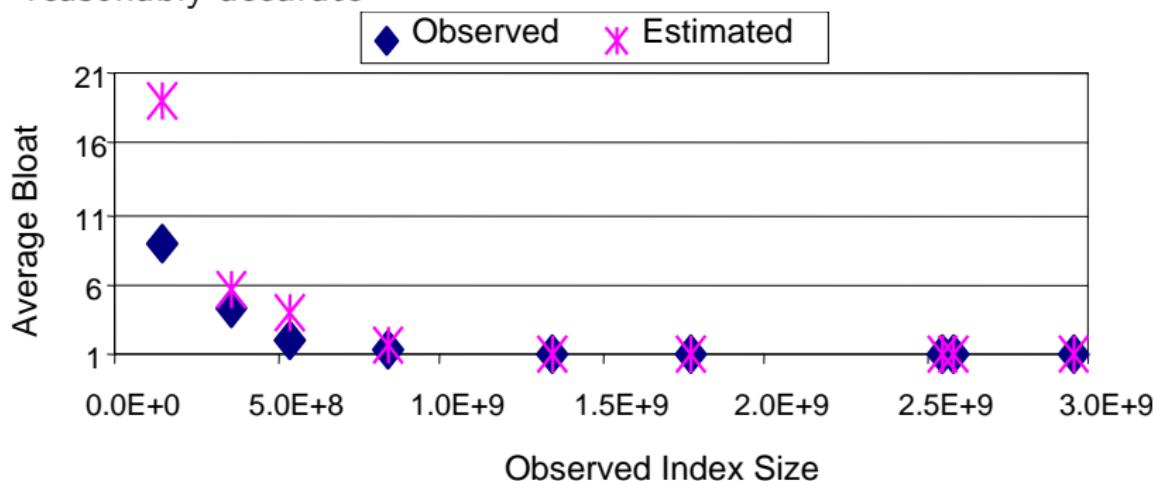
$$\text{exp. bloat} = \sum_{a \in A} \widetilde{\Pr}(a) \frac{t_{\text{scan}} \text{corpusCount}(g) + k' t_{\text{filter}}}{t_{\text{scan}} \text{corpusCount}(a)}$$

Greedy cost-benefit analysis of every type  $a$

## Result of greedy knapsack

- ▶ Only 520 MB index, only 1.9 avg bloat
- ▶ Space comparable to inverted index on stems
- ▶ Estimated query bloat reasonably accurate

Corpus/Index	GBytes
Original corpus	5.72
Gzipped corpus	1.33
Stem index	0.91
Full type A index	4.30
Type subset R index	0.52
Query Bloat	1.90
Reachability index	0.01
Forward index	1.16



## Forward vs. SIP index

- ▶ The “forward” (as against inverted) index is nothing but the corpus itself
- ▶ Pre-digested to tokens, annotations and features
- ▶ Fast access via (document, span) queries
- ▶ Disk seek per candidate to be filtered is impractical
- ▶ Best to squeeze into RAM of a cluster of machines
- ▶ Some forward index access may be possible to replace by packing more info into the inverted index

FINISH

## Dual inversion

- ▶ Document inverted index
  - ▶ Key is type
  - ▶ Posting list has block for each doc
  - ▶ Block has postings
  - ▶ Each posting has position and entity ID
- ▶ Entity inverted index
  - ▶ Key is words/stems as usual
  - ▶ Postings contain doc, pos of stem, [nearby](#) entities, and pos of entity
  - ▶ They use 200 words as “nearby” window — quite large!

## Beyond cached OR postings

- ▶ Main goal: Avoid long list scan/merge using side indices
- ▶ Two main tricks: joint index and contextual index
- ▶ **Joint index:** Pre-merge (in the intersection sense) two or more lists and store co-occurrences within given upper bound window
- ▶ Examples: people and China, in and type:quantity/epoch/year
- ▶ Note, one list can be for a stem and the other for an entity or quantity
- ▶ Which pairs to materialize?
- ▶ Depends on frequencies (length of lists)
- ▶ **Contextual index:** the key is again a single stem or entity (not pair etc.)
- ▶ Posting entry is a doc and pos where stem or entity occurs
- ▶ But with **additional** info inlined into the posting list

## Beyond cached OR postings (2)

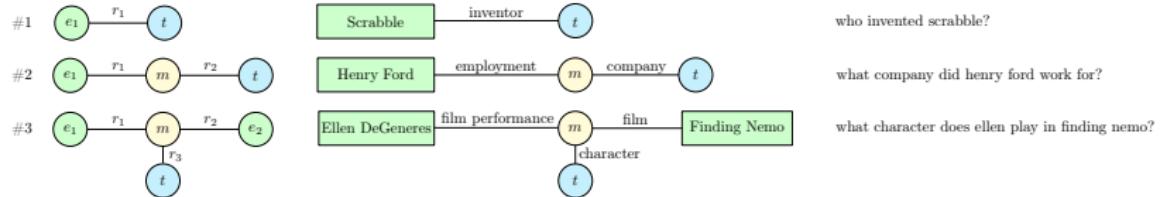
- ▶ Suppose max window size is 5
- ▶ Store inlined all other terms (stems or entities) that occur within 5 tokens of doc and pos
- ▶ Example: document ID “12” has this token sequence:  
... Turkey<sub>(-3)</sub> has<sub>(-2)</sub> a<sub>(-1)</sub> population<sub>(0)</sub> of<sub>(1)</sub> ...
- ▶ Relative offsets from doc and pos shown as subscripts
- ▶ The contextual posting list of “population” will have a posting entry that has
  - ▶ The doc id 12
  - ▶ Absolute position in doc
  - ▶ Left and right context with relative offsets
- ▶ Context term IDs are not very compressible
- ▶ Each token occurrence replicated many times — window better be small!

## Beyond cached OR postings (3)

- ▶ Experimental claim: for window of 5, contextual index is only 2–3 times the size of the standard stem index
- ▶ Important system detail: how to compile queries to take best advantage of joint and contextual indices?

## Query interpretation and question answering

# Role of KG in few-relation QA



- ▶ Input **utterance**  $x$  is a well-formed (parseable) question or “telegraphic” query that seeks an entity or a set/list of entities
- ▶ Target entity or entities belong to knowledge graph (KG  $\mathcal{K}$ )
- ▶ Single relation, or two relations via “complex value type” CVT (aka *mediator*) node
- ▶  $x$  is translated into a **denotation**  $d$ , which is a structured query that can be executed over  $\mathcal{K}$  to retrieve entity response(s)  $y$
- ▶ (Distant) training is via  $(x, y)$  pairs; query denotations  $d$  are not provided
- ▶ A **corpus**  $\mathcal{C}$  may be involved in training the system offline, or choosing the best  $d$  and/or  $y$  at query time

## A barebones baseline [37]

- ▶ 300 lines of Python!
- ▶ Focus on single-relation queries (85% of WebQuestions)
- ▶ First step: find candidates for  $e_1$  grounded in query
  - ▶ No POS tagging, no chunking, fine typing or other NLP stages
  - ▶ Use KG for a large gazette of entity aliases
  - ▶ E.g. query [what character did natalie portman play in star wars?]
  - ▶ Candidate  $e_1$  mentions are character, natalie, natalie\_portman, play, star, and star\_wars
  - ▶ Ranking and pruning?
- ▶ Second step: for each  $e_1$  candidate from KG, predict the relation  $r$  that connects  $e_1$  with an answer  $e_2$ 
  - ▶ Unigram and bigram words from the question are features
  - ▶ Logistic regression to learn a model that predicts relations from features
  - ▶ E.g. query [whats swedens currency?] with gold  $e_2$  = Swedish\_krona
  - ▶ Sweden, /location/country/currency\_used, Swedish\_krona
- ▶ Third step: Report  $e_2$ s where  $(e_1, r, e_2) \in \text{KG}$

## A barebones baseline [37] (2)

- ▶ Given gold  $e_2$  set and system  $e_2$  set, can compute recall, precision and F1 per query
- ▶ Reasonable performance, given extreme simplicity
- ▶ Perhaps more a statement on WebQuestions than techniques

System	F1 <sup>a</sup>	F1 <sup>b</sup>
Yao and Van Durme (2014)	33.0	42.0
Berant and Liang (2014) [14]	39.9	43.0
Reddy et al. (2014)	41.3	-
Bordes et al. (2014)	41.8	45.7
Yao (2015) [37]	44.3	53.5
AQQU, Bast and Haussman (2015) [16]	49.4	-
STAGG, Yih et al. (2015) [15]	52.5	-
Xu et al. (2016) [19]	53.3 <sup>c</sup>	-

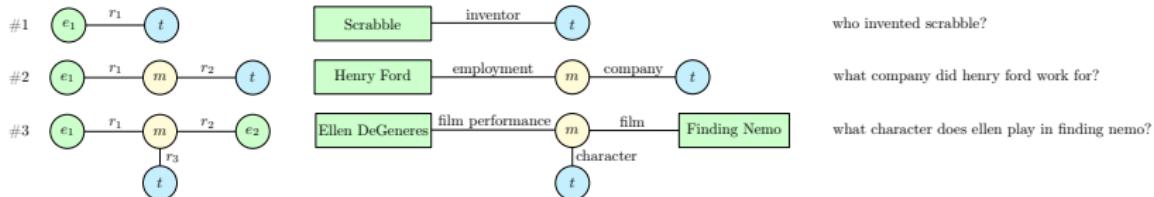
<sup>a</sup>Average F1 over queries

<sup>b</sup>F1 of average R and P over queries, usually larger

<sup>c</sup>Uses corpus-based pruning

# AQQU [16]

- ▶ Works harder on scoring features
- ▶ Supports three query templates



- ▶ First step: entity  $e_1$  matching
- ▶ Second step: candidate (template) generation
- ▶ Relation and answer type matching
- ▶ Third step: feature generation
- ▶ Fourth step: ranking and pruning

## AQQU: Matching $e_1$

- ▶ POS-tag question using Stanford tagger (how good will this be for telegraphic queries?)
- ▶ Consider only NN and NNP for detecting  $e_1$
- ▶ Single-word span must be NN and maximal NNP spans
- ▶ For each span  $s$ , find all KG entities  $e_1$  having  $s$  as an alias
- ▶ Obtained from **CrossWikis** dataset, covering  $\sim 4M$  Wikipedia entities; Freebase aliases more noisy and less complete
- ▶ Another  $\sim 40M$  Freebase entities not in Wikipedia
- ▶ Popularity of each entity  $e$  is the number of times it is mentioned in ClueWeb12
- ▶ CrossWikis already gives an estimated  $\Pr(e|s)$
- ▶ Extrapolate from this to scores for Freebase minus Wikipedia
- ▶ Thus get candidate  $\mathcal{E}_1$  set

## AQQU: Candidate template generation

**Direct edge:** For each  $e_1 \in \mathcal{E}_1$ , find all  $r$  such that  $(e_1, r, e_2) \in \text{KG}$  for some  $e_2$ ; this is a single SPARQL query

**Two-hop:** For each  $e_1 \in \mathcal{E}_1$ , find all  $r_1, m, r_2$  such that  $(e_1, r_1, m), (m, r_2, e_2) \in \text{KG}$  for some  $e_2$ ; two SPARQL queries in sequence

**Three-link star:** For all pairs  $e_1, e'_1 \in \mathcal{E}_1$ , find all  $(e_1, r_1, m), (e'_1, r'_1, m), (m, r_2, e_2) \in \text{KG}$  for some  $e_2$ ; two queries in “inverted indices” to retrieve “posting lists” sorted on  $m$ ; intersect, then issue one SPARQL query to get  $r_2, e_2$  candidates

- ▶ Slightly different search procedure for each of three templates
- ▶ Surely some pruning logic in the code?
- ▶ Why limit later scoring to structured query candidates if we are anyway retrieving  $e_2$ s in this stage?

## AQQU: Relation/s to question text matching

- ▶ Each candidate structured query has one or more of  $r_1, r'_1, r_2$
- ▶ Are there hint spans in the question that justify them?
- ▶ Map each relation to word bag, merge them into RW
- ▶ Likewise, let QW be question words not already matched to  $e_1, e'_1$  etc.
- ▶ **Literal match:** size of overlap
- ▶ **Derivation match:** after conflating (produce, producer), (high, height), (high, elevation)
- ▶ **Embedding match:** Consider synonyms if cosine similarity above a threshold
- ▶ **Context match:** 23 million sentences having  $e_1, e_2$  where  $(e_1, r, e_2)$  holds; find dependency parse; all words on dependency paths thrown into TFIDF bag

## AQQU: Feature generation and ranking candidates

- ▶ Features are generated between the question and a candidate structured query
- ▶ Entity  $e_1, e'_1$  matching features
- ▶ Relation matching features
- ▶ (N-gram relation matching features)
- ▶ 23 features in all
- ▶ From gold  $e_2^*$ , induce gold  $r_1, r'_1, r_2$ , etc.
  - ▶ Small template graphs may be critical here
  - ▶ (How) do they get negative samples?
- ▶ Cast candidate query selection as item- or pair-wise ranking
- ▶ Use logistic regression or random forest

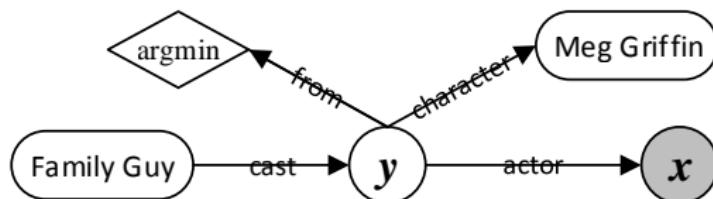
## Hard-segmenting and joint ranking of $e_2$ [27, 18]

- ▶ Be robust to telegraphic questions
- ▶ Do not depend on parsing question
- ▶ Enlist help from corpus as well as KG in a unified model
- ▶ Propose an underlying generative story for the question
- ▶ Aggregate over many possible query interpretations
- ▶ Do not rank and choose interpretations
- ▶ Instead directly rank response entities

Slides from [27, 18] here

## Convnets and staged parsing [15]

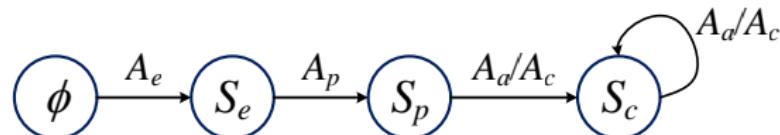
- ▶ Semantic parsing as KG-guided search for query graph generation
- ▶ Three steps: locate  $e_1$ , find main relation (path) between  $e_1$  and answer  $e_2$ , and expand query graph while asserting constraints
- ▶ **Query graph** has four kinds of nodes:
  - Grounded entity:  $e_1, e'_1$ , etc. (rounded rectangle)
  - Existential variable:  $m$  (circle)
  - Lambda variable:  $e_2$  (shaded circle)
  - Aggregation function: constraints (diamond)



Who first voiced Meg on Family Guy?

## Convnets and staged parsing [15] (2)

- ▶ Logical form:  
 $\lambda x. \exists y. \text{cast}(\textit{FamilyGuy}, y) \wedge \text{actor}(y, x) \wedge \text{character}(y, \textit{MegGriffin})$
- ▶ Staging limits exploration of KG and saves time



- ▶  $\mathcal{S} = \emptyset \cup \mathcal{S}_e \cup \mathcal{S}_p \cup \mathcal{S}_c$  is the set of states
  - ▶  $\emptyset$  is an empty graph
  - ▶  $\mathcal{S}_e$  comprises single-node graphs each with an entity  $e_1$  claimed to be grounded in the question
  - ▶  $\mathcal{S}_p$  comprises a **core inferential chain**
  - ▶  $\mathcal{S}_c$  represents additional constraints
- ▶  $\mathcal{A} = \mathcal{A}_e \cup \mathcal{A}_p \cup \mathcal{A}_c \cup \mathcal{A}_a$  is a set of actions
  - ▶  $\mathcal{A}_e$  chooses an entity node
  - ▶  $\mathcal{A}_p$  extends to inferential chain
  - ▶  $\mathcal{A}_c, \mathcal{A}_a$  extend with constraints and aggregators

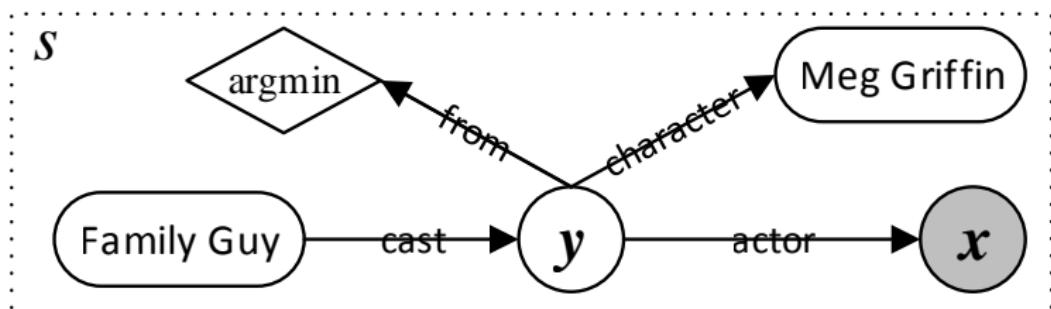
# STAGG: Identifying the core inferential chain



- ▶ Explore legitimate predicate sequences that can start from  $e_1$ 
  - ▶ Explore all paths of length 2 when the middle existential variable can be grounded to a CVT node
  - ▶ Explore paths of length 1 if not
  - ▶ Some longer paths allowed, if observed in training data 😊
- ▶ By now, query is [Who first voiced    on   ?]
- ▶ Choose from cast-actor, writer-start, genre, ...
- ▶ Measuring semantic similarity using (Siamese?) convnets
  - ▶ One convnet processes (residual) query
  - ▶ The other processes edge label sequences (too short for conv?)

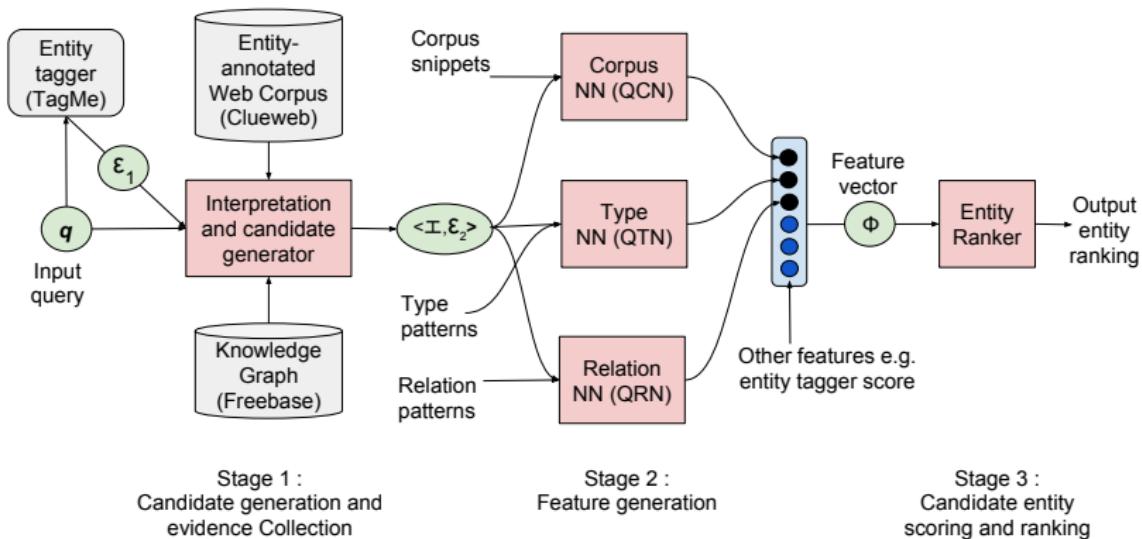
## STAGG: Scoring progress of staged parsing

$q$  = “Who first voiced Meg on Family Guy?”



- (1) EntityLinkingScore(FamilyGuy, “Family Guy”) = 0.9
- (2) PatChain(“who first voiced meg on <e>”, cast-actor) = 0.7
- (3) QuesEP( $q$ , “family guy cast-actor”) = 0.6
- (4) ClueWeb(“who first voiced meg on <e>”, cast-actor) = 0.2
- (5) ConstraintEntityWord(“Meg Griffin”,  $q$ ) = 0.5
- (6) ConstraintEntityInQ(“Meg Griffin”,  $q$ ) = 1
- (7) AggregationKeyword(argmin,  $q$ ) = 1
- (8) NumNodes(s) = 5
- (9) NumAns(s) = 1

# Simpler 3-convnet system



- ▶ 2+(1+1) convnets: three run on question (minus  $e_1$ ), one runs on corpus snippets
- ▶ Goal is to score not structured query but  $e_2$  directly

## References

- [1] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan, "Keyword searching and browsing in databases using BANKS," in *ICDE*. IEEE, 2002.
- [2] S. Chakrabarti, "Dynamic personalized PageRank in entity-relation graphs," in *WWW Conference*, Banff, May 2007. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/netrank/>
- [3] H. Garcia-Molina, R. Goldman, N. Shivakumar, and S. Venkatasubramanian, "Proximity search in databases," in *VLDB Conference*, 1998.
- [4] D. Florescu, D. Kossmann, and I. Manolescu, "Integrating keyword searches into XML query processing," in *WWW Conference*, Amsterdam, 2000, pp. 119–135. [Online]. Available: <http://www9.org/w9cdrom/324/324.html>
- [5] T. T. Chinenyanga and N. Kushmerick, "An expressive and efficient language for XML information retrieval," *JASIST*, vol. 53, no. 6, pp. 438–453, 2002. [Online]. Available: <citeseer.ist.psu.edu/andn01expressive.html>

## References (2)

- [6] N. Fuhr and K. Grosjohann, "XIRQL: A query language for information retrieval in XML documents," in *SIGIR Conference*, 2001, pp. 172–180. [Online]. Available: [http://www.is.informatik.uni-duisburg.de/bib/pdf/ir/Fuhr\\_Grossjohann:01.pdf](http://www.is.informatik.uni-duisburg.de/bib/pdf/ir/Fuhr_Grossjohann:01.pdf)
- [7] S. Agrawal, S. Chaudhuri, and G. Das, "DBXplorer: A system for keyword-based search over relational databases," in *ICDE*. San Jose, CA: IEEE, 2002.
- [8] A. Balmin, V. Hristidis, and Y. Papakonstantinou, "Authority-based keyword queries in databases using ObjectRank," in *VLDB Conference*, Toronto, 2004.
- [9] T. Cheng, X. Yan, and K. C.-C. Chang, "EntityRank: Searching entities directly and holistically," in *VLDB Conference*, Sep. 2007, pp. 387–398. [Online]. Available: <http://www-forward.cs.uiuc.edu/pubs/2007/entityrank-vldb07-cyc-jul07.pdf>

## References (3)

- [10] S. Chakrabarti, K. Punyani, and S. Das, "Optimizing scoring functions and indexes for proximity search in type-annotated corpora," in *WWW Conference*, Edinburgh, May 2006, pp. 717–726. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/www2006i>
- [11] P. Sarkar, A. W. Moore, and A. Prakash, "Fast incremental proximity search in large graphs," in *ICML*, 2008, pp. 896–903. [Online]. Available: <http://icml2008.cs.helsinki.fi/papers/565.pdf>
- [12] G. Kasneci, F. M. Suchanek, G. Ifrim, S. Elbassuoni, M. Ramanath, and G. Weikum, "NAGA: harvesting, searching and ranking knowledge," in *SIGMOD Conference*. ACM, 2008, pp. 1285–1288. [Online]. Available: <http://www.mpi-inf.mpg.de/~kasneci/naga/>
- [13] G. Kasneci, F. M. Suchanek, G. Ifrim, M. Ramanath, and G. Weikum, "NAGA: Searching and ranking knowledge," in *ICDE*. IEEE, 2008.

## References (4)

- [14] J. Berant, A. Chou, R. Frostig, and P. Liang, "Semantic parsing on Freebase from question-answer pairs," in *EMNLP Conference*, 2013, pp. 1533–1544. [Online]. Available: <http://aclweb.org/anthology//D/D13/D13-1160.pdf>
- [15] S. W.-t. Yih, M.-W. Chang, X. He, and J. Gao, "Semantic parsing via staged query graph generation: Question answering with knowledge base," in *ACL Conference*, 2015, pp. 1321–1331. [Online]. Available: <http://anthology.aclweb.org/P/P15/P15-1128.pdf>
- [16] H. Bast and E. Haußmann, "More accurate question answering on freebase," in *CIKM*, 2015, pp. 1431–1440. [Online]. Available: [http://ad-publications.informatik.uni-freiburg.de/CIKM\\_freebase\\_qa\\_BH\\_2015.pdf](http://ad-publications.informatik.uni-freiburg.de/CIKM_freebase_qa_BH_2015.pdf)
- [17] Y. Fang, L. Si, and A. P. Mathur, "Discriminative models of integrating document evidence and document-candidate associations for expert search," in *SIGIR Conference*, 2010. [Online]. Available: [http://www.cs.purdue.edu/homes/fangy/SIGIR2010\\_Expert\\_Search.pdf](http://www.cs.purdue.edu/homes/fangy/SIGIR2010_Expert_Search.pdf)

## References (5)

- [18] M. Joshi, U. Sawant, and S. Chakrabarti, "Knowledge graph and corpus driven segmentation and answer inference for telegraphic entity-seeking queries." in *EMNLP Conference*, 2014, pp. 1104–1114, download <http://bit.ly/1OCKbVW>. [Online]. Available: <http://www.emnlp2014.org/papers/pdf/EMNLP2014117.pdf>
- [19] K. Xu, S. Reddy, Y. Feng, S. Huang, and D. Zhao, "Question answering on Freebase via relation extraction and textual evidence," *arXiv preprint arXiv:1603.00957*, 2016. [Online]. Available: <https://arxiv.org/pdf/1603.00957.pdf>
- [20] D. Savenkov and E. Agichtein, "When a knowledge base is not enough: Question answering over knowledge bases with external text data," in *SIGIR Conference*, 2016, pp. 235–244. [Online]. Available: <https://dl.acm.org/citation.cfm?id=2911536>
- [21] W. W. Cohen, "Integration of heterogeneous databases without common domains using queries based on textual similarity," in *SIGMOD Conference*. Seattle, Washington: ACM, 1998.

## References (6)

- [22] V. Hristidis, L. Gravano, and Y. Papakonstantinou, "Efficient IR-style keyword search over relational databases," in *VLDB Conference*, 2003, pp. 850–861. [Online]. Available: <http://www.db.ucsd.edu/publications/VLDB2003cr.pdf>
- [23] S. Chakrabarti, A. Pathak, and M. Gupta, "Index design and query processing for graph conductance search," *VLDB Journal*, 2010. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/HubRank/>
- [24] A. Agarwal, S. Chakrabarti, and S. Aggarwal, "Learning to rank networked entities," in *SIGKDD Conference*, 2006, pp. 14–23. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/netrank>
- [25] S. Banerjee, S. Chakrabarti, and G. Ramakrishnan, "Learning to rank for quantity consensus queries," in *SIGIR Conference*, 2009. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/QCQ/>

## References (7)

- [26] X. Yao and B. Van Durme, "Information extraction over structured data: Question answering with Freebase," in *ACL Conference*. ACL, 2014. [Online]. Available:  
<http://www.cs.jhu.edu/~xuchen/paper/yao-jacana-freebase-acl2014.pdf>
- [27] U. Sawant and S. Chakrabarti, "Learning joint query interpretation and response ranking," in *WWW Conference*, Brazil, 2013. [Online]. Available:  
<http://arxiv.org/abs/1212.6193>
- [28] Z. Sun, H. Wang, H. Wang, B. Shao, and J. Li, "Efficient subgraph matching on billion node graphs," *PVLDB*, vol. 5, no. 9, pp. 788–799, 2012. [Online]. Available: <https://arxiv.org/pdf/1205.6691>
- [29] P. P. Talukdar, M. Jacob, M. S. Mahmood, K. Crammer, Z. G. Ives, F. Pereira, and S. Guha, "Learning to create data-integrating queries," *PVLDB*, vol. 1, no. 1, pp. 785–796, 2008. [Online]. Available:  
<http://www.vldb.org/pvldb/1/1453941.pdf>

## References (8)

- [30] B. Dalvi, M. Kshirsagar, and S. Sudarshan, "Keyword search on external memory data graphs," in *VLDB Conference*, 2008. [Online]. Available: <http://www.cse.iitb.ac.in/~sudarsha/>
- [31] H. He, H. Wang, J. Yang, and P. S. Yu, "BLINKS: Ranked keyword searches on graphs," in *SIGMOD Conference*. New York, NY, USA: ACM, 2007, pp. 305–316.
- [32] Y. Lv and C. Zhai, "Positional language models for information retrieval," in *SIGIR Conference*, 2009, pp. 299–306. [Online]. Available: <http://sifaka.cs.uiuc.edu/czhai/pub/sigir09-PLM.pdf>
- [33] D. Petkova and W. B. Croft, "Proximity-based document representation for named entity retrieval," in *CIKM*. ACM, 2007, pp. 731–740. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1321440.1321542>
- [34] K. M. Svore, P. H. Kanani, and N. Khan, "How good is a span of terms? exploiting proximity to improve Web retrieval," in *SIGIR Conference*. ACM, 2010, pp. 154–161. [Online]. Available: <http://research.microsoft.com/apps/pubs/default.aspx?id=121723>

## References (9)

- [35] J. Ko, E. Nyberg, and L. Si, "A probabilistic graphical model for joint answer ranking in question answering," in *SIGIR Conference*, 2007, pp. 343–350. [Online]. Available:  
<http://www.cs.cmu.edu/~jko/paper/sigir.pdf>
- [36] T. Qin, T.-Y. Liu, X.-D. Zhang, D.-S. Wang, W.-Y. Xiong, and H. Li, "Learning to rank relational objects and its application to Web search," in *WWW Conference*, 2008, pp. 407–416. [Online]. Available:  
<http://www2008.org/papers/pdf/p407-qinA.pdf>
- [37] X. Yao, "Lean question answering over Freebase from scratch," in *NAACL Conference*, 2015, pp. 66–70. [Online]. Available:  
[http://www.aclweb.org/website/old\\_anthology/N/N15/N15-3014.pdf](http://www.aclweb.org/website/old_anthology/N/N15/N15-3014.pdf)

## Timeline: some highlights

- ∞ Hidden Markov models
- 1992 Hearst patterns [1]
- 1998 Duality in pattern-relationship extraction [2]
- 2000 Snowball = DIPRE + confidence scores [3]
- 2001 Conditional Random Fields [4]
- 2001 Turney's PMI [5]
- 2001 SemTag and Seeker [6]
- 2000–now Many systems for labeling token spans [4] or 2d regions [7] with entity types
- 2001–2002 Search in graph data models [8, 9, 10]
- 2002–2003 Personalized and topic-specific PageRank[11, 12]
- 2004 OBJECTRANK[13]

## Timeline: some highlights (2)

- 2004 KnowItAll  $\approx$  Hearst patterns + list extraction + a few more tricks [14]
- 2005 Relation extraction via dependency path kernels [15]
- 2007 TextRunner [16] and ExDB [17]
- 2006–2007 Type+proximity search, ENTITYRANK[18, 19]
- 2007–2008 Proximity search in graphs [20, 21]
- 2006–2009 Entity disambiguation [22, 23, 24, 25, 26]
- 2007–2009 Searching the structured-unstructured divide [27, 28]
- 2010– Never ending language learning (NELL, @CMU)
- 2011–2012 MultiR, MIML-RE [29]
- 2013–2014 Word2vec [30], GloVE [31]
- 2014– Continuous knowledge representation for KBC [32]
- 2011, 2015 Path-ranking algorithm, KG+corpus for relation extraction [33, 34]
- 2004–2006, 2009, 2013–2016 KG+corpus in QA

## References

- [1] M. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *International Conference on Computational Linguistics*, vol. 14, 1992, pp. 539–545. [Online]. Available: [http://www.aclweb.org/website/old\\_anthology/C/C92/C92-2082.pdf](http://www.aclweb.org/website/old_anthology/C/C92/C92-2082.pdf)
- [2] S. Brin, "Extracting patterns and relations from the World Wide Web," in *WebDB Workshop*, ser. LNCS, P. Atzeni, A. O. Mendelzon, and G. Mecca, Eds., vol. 1590. Valencia, Spain: Springer, Mar. 1998, pp. 172–183. [Online]. Available: <http://ilpubs.stanford.edu:8090/421/1/1999-65.pdf>
- [3] E. Agichtein and L. Gravano, "Snowball: Extracting relations from large plain-text collections," in *ICDL*, 2000, pp. 85–94. [Online]. Available: <http://www.academia.edu/download/31007490/cucs-033-99.pdf>
- [4] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *ICML*, 2001, pp. 282–289. [Online]. Available: [ftp://ftp.cis.upenn.edu/pub/datamining/public\\_html/ReadingGroup/papers/crf.pdf](ftp://ftp.cis.upenn.edu/pub/datamining/public_html/ReadingGroup/papers/crf.pdf)

## References (2)

- [5] P. D. Turney, "Mining the Web for synonyms: PMI-IR versus LSA on TOEFL," in *ECML*, 2001.
- [6] S. Dill *et al.*, "SemTag and Seeker: Bootstrapping the semantic Web via automated semantic annotation," in *WWW Conference*, 2003, pp. 178–186.
- [7] J. Zhu, Z. Nie, B. Zhang, and J.-R. Wen, "Dynamic hierarchical Markov random fields and their application to Web data extraction," in *ICML*, 2007, pp. 1175–1182. [Online]. Available: <http://www.machinelearning.org/proceedings/icml2007/papers/215.pdf>
- [8] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan, "Keyword searching and browsing in databases using BANKS," in *ICDE*. IEEE, 2002.
- [9] S. Agrawal, S. Chaudhuri, and G. Das, "DBXplorer: A system for keyword-based search over relational databases," in *ICDE*. San Jose, CA: IEEE, 2002.

## References (3)

- [10] V. Hristidis, L. Gravano, and Y. Papakonstantinou, "Efficient IR-style keyword search over relational databases," in *VLDB Conference*, 2003, pp. 850–861. [Online]. Available:  
<http://www.db.ucsd.edu/publications/VLDB2003cr.pdf>
- [11] G. Jeh and J. Widom, "Scaling personalized web search," in *WWW Conference*, 2003, pp. 271–279. [Online]. Available:  
<https://goo.gl/oAZZLq>
- [12] T. H. Haveliwala, "Topic-sensitive PageRank," in *WWW Conference*, 2002, pp. 517–526. [Online]. Available:  
<http://www2002.org/CDROM/refereed/127/index.html>
- [13] A. Balmin, V. Hristidis, and Y. Papakonstantinou, "Authority-based keyword queries in databases using ObjectRank," in *VLDB Conference*, Toronto, 2004.

## References (4)

- [14] O. Etzioni, M. Cafarella *et al.*, "Web-scale information extraction in KnowItAll," in *WWW Conference*. New York: ACM, 2004. [Online]. Available: <http://www.cs.washington.edu/research/knowitall/papers/www-paper.pdf>
- [15] R. C. Bunescu and R. J. Mooney, "A shortest path dependency kernel for relation extraction," in *EMNLP Conference*. ACL, 2005, pp. 724–731. [Online]. Available: <http://acl.ldc.upenn.edu/H/H05/H05-1091.pdf>
- [16] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open information extraction from the Web," in *IJCAI*, M. M. Veloso, Ed., 2007, pp. 2670–2676. [Online]. Available: <http://www.ijcai.org/papers07/Papers/IJCAI07-429.pdf>
- [17] M. J. Cafarella, C. Re, D. Suciu, O. Etzioni, and M. Banko, "Structured querying of web text: A technical challenge," in *CIDR*, 2007, pp. 225–234. [Online]. Available: <http://www-db.cs.wisc.edu/cidr/cidr2007/papers/cidr07p25.pdf>

## References (5)

- [18] S. Chakrabarti, K. Punyani, and S. Das, "Optimizing scoring functions and indexes for proximity search in type-annotated corpora," in *WWW Conference*, Edinburgh, May 2006, pp. 717–726. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/www2006i>
- [19] T. Cheng, X. Yan, and K. C.-C. Chang, "EntityRank: Searching entities directly and holistically," in *VLDB Conference*, Sep. 2007, pp. 387–398. [Online]. Available: <http://www-forward.cs.uiuc.edu/pubs/2007/entityrank-vldb07-cyc-jul07.pdf>
- [20] S. Chakrabarti, "Dynamic personalized PageRank in entity-relation graphs," in *WWW Conference*, Banff, May 2007. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/netrank/>
- [21] P. Sarkar, A. W. Moore, and A. Prakash, "Fast incremental proximity search in large graphs," in *ICML*, 2008, pp. 896–903. [Online]. Available: <http://icml2008.cs.helsinki.fi/papers/565.pdf>

## References (6)

- [22] R. Bunescu and M. Pasca, "Using encyclopedic knowledge for named entity disambiguation," in *EACL*, 2006, pp. 9–16. [Online]. Available: <http://www.cs.utexas.edu/~ml/papers/encyc-eacl-06.pdf>
- [23] R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *CIKM*, 2007, pp. 233–242. [Online]. Available: <http://portal.acm.org/citation.cfm?id=1321440.1321475>
- [24] S. Cucerzan, "Large-scale named entity disambiguation based on Wikipedia data," in *EMNLP Conference*, 2007, pp. 708–716. [Online]. Available: <http://www.aclweb.org/anthology/D/D07/D07-1074>
- [25] D. Milne and I. H. Witten, "Learning to link with Wikipedia," in *CIKM*, 2008, pp. 509–518. [Online]. Available: <http://www.cs.waikato.ac.nz/~nk2/publications/CIKM08-LearningToLinkWithWikipedia.pdf>
- [26] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti, "Collective annotation of Wikipedia entities in Web text," in *SIGKDD Conference*, 2009, pp. 457–466. [Online]. Available: <http://www.cse.iitb.ac.in/~soumen/doc/CSAW/>

## References (7)

- [27] G. Kasneci, F. M. Suchanek, G. Ifrim, S. Elbassuoni, M. Ramanath, and G. Weikum, "NAGA: harvesting, searching and ranking knowledge," in *SIGMOD Conference*. ACM, 2008, pp. 1285–1288. [Online]. Available: <http://www.mpi-inf.mpg.de/~kasneci/naga/>
- [28] F. M. Suchanek, G. Kasneci, and G. Weikum, "YAGO: A core of semantic knowledge unifying WordNet and Wikipedia," in *WWW Conference*. ACM Press, 2007, pp. 697–706. [Online]. Available: <http://www2007.org/papers/paper391.pdf>
- [29] M. Surdeanu, J. Tibshirani, R. Nallapati, and C. D. Manning, "Multi-instance multi-label learning for relation extraction," in *EMNLP Conference*, 2012, pp. 455–465. [Online]. Available: <http://anthology.aclweb.org/D/D12/D12-1042.pdf>
- [30] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS Conference*, 2013, pp. 3111–3119. [Online]. Available: <https://goo.gl/x3DTzS>

## References (8)

- [31] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation." in *EMNLP Conference*, vol. 14, 2014, pp. 1532–1543. [Online]. Available: <http://www.emnlp2014.org/papers/pdf/EMNLP2014162.pdf>
- [32] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *NIPS Conference*, 2013, pp. 2787–2795. [Online]. Available: <http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data.pdf>
- [33] N. Lao and W. W. Cohen, "Relational retrieval using a combination of path-constrained random walks," *Machine Learning*, vol. 81, no. 1, pp. 53–67, Oct. 2010. [Online]. Available: <http://dx.doi.org/10.1007/s10994-010-5205-8>
- [34] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," in *EMNLP Conference*, 2015, pp. 1499–1509. [Online]. Available: <https://www.aclweb.org/anthology/D/D15/D15-1174.pdf>

## References (9)

- [35] S. Sarawagi, "Information extraction," *FnT Databases*, vol. 1, no. 3, 2008. [Online]. Available:  
<http://www.cse.iitb.ac.in/~sunita/papers/ieSurvey.pdf>



The End