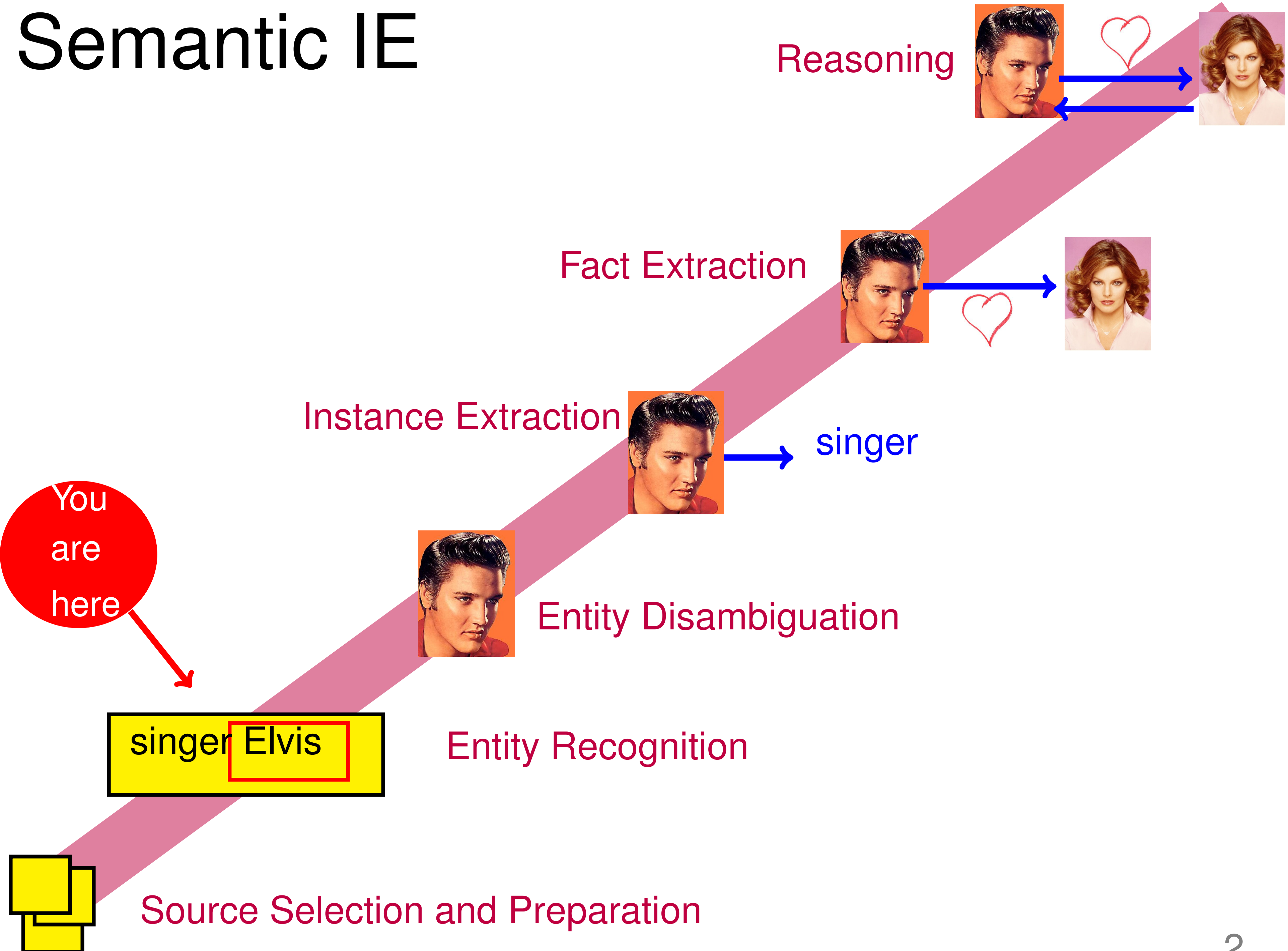
Named Entity Annotation

Fabian M. Suchanek

Semantic IE


Reasoning
Fact Extraction
Instance Extraction
singer
You are here
Entity Disambiguation
singer Elvis
Entity Recognition
Source Selection and Preparation

Overview

- Named Entity Annotation (NEA)
- NEA by rules
  - Learning NEA rules
- NEA by statistical models
  - Learning statistical NEA models

Def: Named Entity Annotation

Named Entity Annotation (NEA) is the task of (1) finding entity names in a corpus and (2) annotating each name with a class out of a set of given classes.

(This is commonly known as “Named Entity Recognition”. The name “NEA” is introduced here to distinguish NER with classes from NER without classes)

classes={Person, Location, Organization}

Ford Prefect eats at Milliways.

Def: Named Entity Annotation

Named Entity Annotation (NEA) is the task of (1) finding entity names in a corpus and (2) annotating each name with a class out of a set of given classes.

(This is commonly known as “Named Entity Recognition”. The name “NEA” is introduced here to distinguish NER with classes from NER without classes)

classes={Person, Location, Organization}

Ford Prefect eats at Milliways.

Person

Organization

Def: Named Entity Annotation

Named Entity Annotation (NEA) is the task of (1) finding entity names in a corpus and (2) annotating each name with a class out of a set of given classes.

(This is commonly known as "Named Entity Recognition". The name "NEA" is introduced here to distinguish NER with classes from NER without classes)

classes={Person, Location, Organization}

Ford Prefect eats at Milliways.

Person

Organization

NEA usually focuses on
- person
- location
- organization

...but there are also systems that extract money, percent, phone number, job title, artefact, brand, product, protein, drug, etc.

NEA examples

Arthur Dent visits Milliways, a restaurant
located at End of the Universe Street 42.

<per>Arthur Dent</per> visits <org>Milliways</org>, a restaurant
located at <loc>End of the Universe Street 42</loc>.

Try this

Now do it here:

We have determined the crystal structure of a triacylglycerol lipase from Pseudomonas cepacia (Pet) in the absence of a bound inhibitor using X-ray crystallography. The structure shows the lipase to contain an alpha/beta-hydrolase fold and a catalytic triad comprising of residues Ser87, His286 and Asp264. The enzyme shares ...

NEA is not easy

- Organization vs. Location

  England won the World Cup.

  The World Cup took place in England.

- Company vs Artefact

  shares in MTV

  watching MTV

- Location vs. Organization

  she met him at Heathrow

  the Heathrow authorities

Overview

- Named Entity Annotation (NEA)
- NEA by rules
  - Learning NEA rules
- NEA by statistical models
  - Learning statistical NEA models

Def: NEA by rules

NEA by rules uses rules of the form

PATTERN => CLASS

It annotates a string by with CLASS, if it matches the PATTERN.

(Patterns and class annotation types can take various forms)

([A-Z][a-z]+) (City—Forest) => Location

Light City is the only city in Ursa Minor.

Location

Def: NEA Pattern, NEA Feature

A NEA pattern is a sequence of features.

A feature is a string with an associated language.

(Patterns are called “Contextual Patterns”, and features are called “labeled patterns” in Sarawagi’s survey.
“Language” is a set of strings, as for regular expressions.)

$\underbrace{\text{([A-Z][a-z]+)}}_{\text{Feature}}\ \underbrace{\text{(City—Forest)}}_{\text{Feature}}$ => Location

Language: {Aa,Ab,Aab,...}

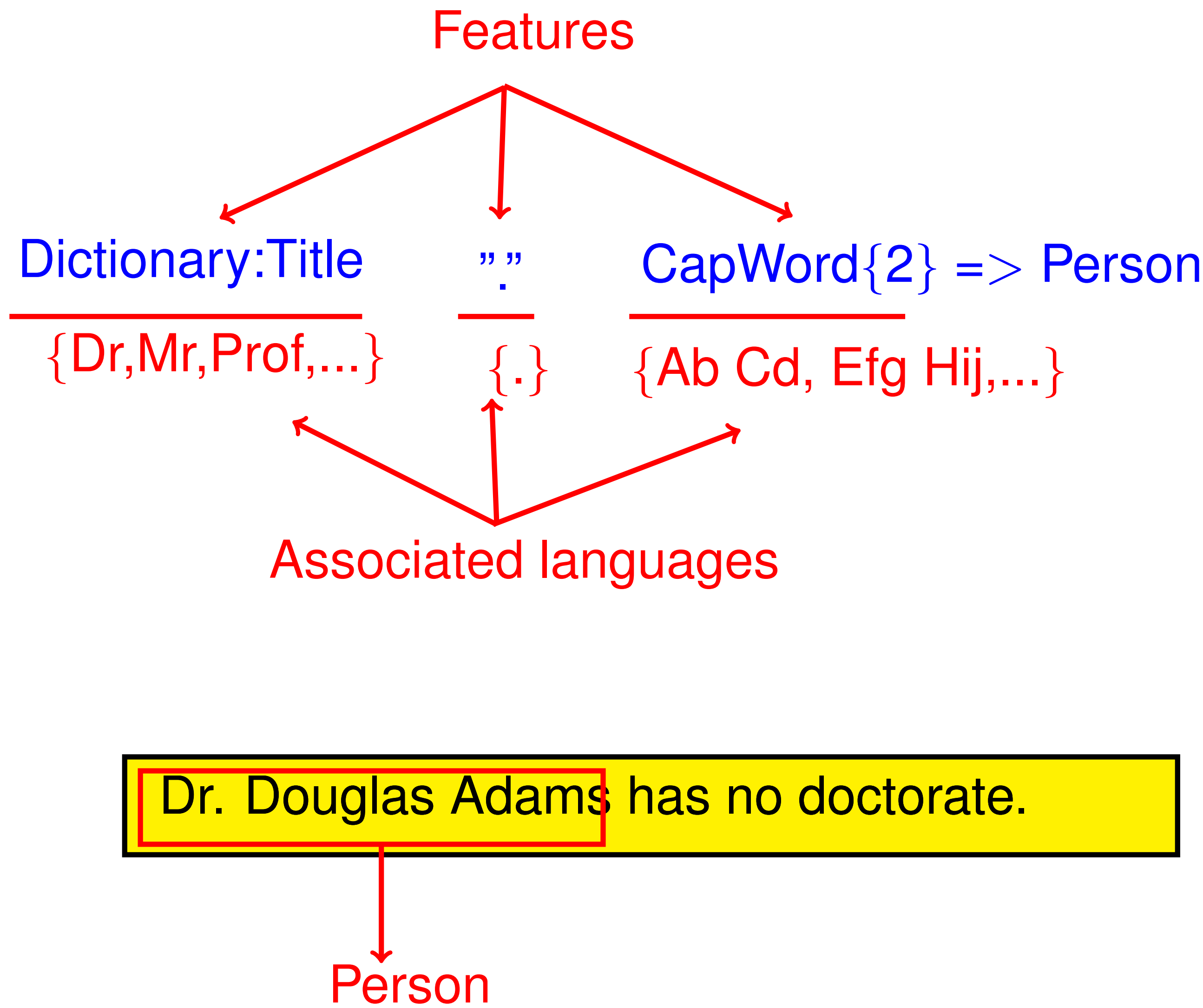
Language: {City,Forest}

Features can be more than regexes

Features

Dictionary:Title ”.” CapWord{2} => Person

Features can be more than regexes



Features, their syntax, and their language are system dependent.

Features

• capitalized word	Fenchurch
• all upper case word	WSOGMM
• smallcase word	planet
• mixed letters/numbers	HG2G
• number	42
• special symbol	
• space	
• punctuation	.,;:?!

• cities	Vassilian
• countries	UK
• titles	Dr.
• common names	Arthur
• airport codes	CDG
• words that identify a company	Inc, Corp, ...

Linguistic features

- word type — verb, noun, ...
- word endings — -ish, -ist, ...
- common noun — car, president, ...
  (helpful to distinguish named entities at the beginning of a sentence)

Def: Context/Designated features

Designated features in a NEA pattern are those whose match will be annotated. The others are context features.

(There is no standard terminology and no standard syntax)

”a pub in” [CapWord] => location

Arthur and Fenchurch have a drink in a pub in Taunton.

<loc>Taunton</loc>

Examples:

- CapWord (Street—Av—Road) => Location
- ”based in” [CapWord] => Location
- ”to the” Compass ”of” [CapWord] => Loc

NEA examples from GATE

286 *Entity Extraction: Rule-based Methods*

Rule: TheGazOrganization
Priority: 50
// Matches "The <in list of company names>"
({Part of speech = DT | Part of speech = RB} {DictionaryLookup = organization})
→ Organization

Rule: LocOrganization
Priority: 50
// Matches "London Police"
({DictionaryLookup = location | DictionaryLookup = country} {DictionaryLookup = organization} {DictionaryLookup = organization}?) → Organization

Rule: INOrgXandY
Priority: 200
// Matches "in Bradford & Bingley", or "in Bradford & Bingley Ltd"
({Token string = "in"})
({Part of speech = NNP}+ {Token string = "&"} {Orthography type = upperInitial}+ {DictionaryLookup = organization end}?):orgName → Organization=:orgName

Rule: OrgDept
Priority: 25
// Matches "Department of Pure Mathematics and Physics"
({Token.string = "Department"} {Token.string = "of"} {Orthography type = upperInitial}+ ({Token.string = "and"} {Orthography type = upperInitial}+)?) → Organization

task>2

Task: NEA patterns

Design NEA patterns that can find planets in the following text. Describe each feature.

(Patterns should generalize beyond the names in this text.)

Lamuella is the nice planet where Arthur Dent lives.
Santraginus V is a planet with marble-sanded beaches.
Magrathea is an ancient planet in Nebula.
The fifty-armed Jatravartids live on Viltvodle VI.

Example:

[CapWord] ”is a planet”

(this pattern does not work, it’s here for inspiration)

Possible Solution: NEA patterns

[CapWord] ”is” (the—a—an) Adj ”planet”

- CapWord: A word that starts with a capital letter.
- ”is”, ”planet”: plain strings
- (the—a—an): the words ”the”, ”a”, or ”an”
- Adj: an adjective (dictionary lookup)

[CapWord RomanNumeral]

- CapWord: as above
- RomanNumeral: A roman numeral (I, II, V, X, ...)

Conflicting NEA rules

If two NEA rules match overlapping strings, we have to decide which one to apply. Possible strategies:

- annotate only with the rule that has a longer match
- manually define order of precedence

CapWord CapWord RomanNum => planet

CapWord ”Minor” => planet

He lives on Ursa Minor IV.

cascade>28

Def: Cascaded NEA

Cascaded NEA applies NEA to the corpus annotated by a previous NEA run.

Main Street 42, West City

↓ First NEA run

<street>Main Street 42</street>, <city>West City</city>

↓ Second First NEA run

<adr><street>Main Street 42</street>, <city>West City</city></adr>

Cascaded NEA rule:

[Street City] => Adr

- Street: a previously annotated street
- City: a previously annotated city

>task

Task: Cascaded NEA

Write NEA rules for the first run and the second, cascaded run of a NEA to recognize person names as in

Dr. Bob Miller
Monsieur François Hollande
Mademoiselle Alizée Jacozey
Ms Gary Day-Ellison

Possible Solution: Cascaded NEA

First run:

Dictionary:AcademicTitle => Title

Dictionary:FrenchTitle => Title

Dictionary:EnglishTitle => Title

CapWord-CapWord => Name

CapWord => Name

Second run:

Title Name Name => Person

Matching NEA rules

Given a NEA rule and a corpus, how can we match the rule on the corpus?

One possibility is to compile the rule to a regular expression:

Dict:Title [CapWord] => Person

↓

(Dr—Prof—Mr—Ms) [A-Z][a-z]+ => Person

Overview

- Named Entity Annotation (NEA)
- NEA by rules
  - Learning NEA rules
- NEA by statistical models
  - Learning statistical NEA models

Learning NEA Rules

NEA rules are usually designed manually (as in the GATE system).
However, they can also be learned automatically
(as in the Rapier, LP2, FOIL, and WHISK systems).

We will now see a blueprint for a bottom-up
rule learning algorithm.

learn>43

Example: Rule learning

0. Start with annotated training corpus

<pers>Arthur</pers> says “Hello”

Example: Rule learning

0. Start with annotated training corpus

1. Find a NEA rule for each annotation

<pers>Arthur</pers> says “Hello”

[Arthur] ”says “Hello”” => pers

Example: Rule learning

0. Start with annotated training corpus

<pers>Arthur</pers> says “Hello”

1. Find a NEA rule for each annotation

[Arthur] ”says “Hello”” => pers

[Ford] ”says “Hello”” => pers

Example: Rule learning

0. Start with annotated training corpus

<pers>Arthur</pers> says “Hello”

1. Find a NEA rule for each annotation

[Arthur] ”says “Hello”” => pers

[Ford] ”says “Hello”” => pers

2. Merge two rules by replacing a feature by a more general feature

Generalize

[CapWord] ”says “Hello”” => pers

Example: Rule learning

0. Start with annotated training corpus

<pers>Arthur</pers> says “Hello”

1. Find a NEA rule for each annotation

[Arthur] ”says “Hello”” => pers

[Ford] ”says “Hello”” => pers

2. Merge two rules by replacing a feature by a more general feature

Generalize

[CapWord] ”says “Hello”” => pers

[CapWord] ”says “Bye”” => pers

Example: Rule learning

0. Start with annotated training corpus

1. Find a NEA rule for each annotation

2. Merge two rules by replacing a feature by a more general feature

3. Merge two rules by dropping a feature

<pers>Arthur</pers> says “Hello”

[Arthur] ”says “Hello””” => pers

[Ford] ”says “Hello””” => pers

Generalize

[CapWord] ”says “Hello””” => pers

[CapWord] ”says “Bye””” => pers

Drop

[CapWord] ”says” => pers

Example: Rule learning

0. Start with annotated
   training corpus

<pers>Arthur</pers> says “Hello”

1. Find a NEA rule
   for each annotation

[Arthur] ”says “Hello”” => pers

[Ford] ”says “Hello”” => pers

2. Merge two rules by replacing
   a feature by a more general
   feature

Generalize

[CapWord] ”says “Hello”” => pers

[CapWord] ”says “Bye”” => pers

3. Merge two rules by dropping
   a feature

Drop

[CapWord] ”says” => pers

[CapWord] (says—yells—screams)=>pe

Example: Rule learning

0. Start with annotated training corpus

<pers>Arthur</pers> says “Hello”

1. Find a NEA rule for each annotation

[Arthur] ”says “Hello”” => pers

[Ford] ”says “Hello”” => pers

2. Merge two rules by replacing a feature by a more general feature

Generalize

[CapWord] ”says “Hello”” => pers

[CapWord] ”says “Bye”” => pers

3. Merge two rules by dropping a feature

Drop

~~[CapWord] ”says” => pers~~

[CapWord] (says—yells—screams)=>pe

4. Remove redundant rules
5. Repeat

NEA rule learning is not easy

There are exponentially many ways to merge rules.

Then [Ford] ”says ’Hello’” => pers

And [Arthur] ”yells ’Bye’” => pers

NEA rule learning is not easy

There are exponentially many ways to merge rules.

Then [Ford] ”says ’Hello’” => pers

And [Arthur] ”yells ’Bye’” => pers

Conj

Cap

Word

Drop

NEA rule learning is not easy

There are exponentially many ways to merge rules.

Then [Ford] ”says ’Hello’” => pers

And [Arthur] ”yells ’Bye’” => pers

Conj	Name	Verb	String
Cap	Cap	(s.—y.)	(Hello—Bye)
Word	Word	Word	Word
Drop	1stNam	Drop	Drop

Goal of NEA rule learning

Learn rules that

- cover all annotations
- don’t cover non-annotated strings
- are not too specific/numerous

  (we do not want 1 rule for each annotation)

Overview

- Named Entity Annotation (NEA)
- NEA by rules
  - Learning NEA rules
- $\boxed{\text{NEA by statistical models}}$
  - Learning statistical NEA models

Def: Statistical NEA corpus

A corpus for statistical NEA is a vector of words (“tokens”).

The output is a vector of class names.

(Words that fall into no class are annotated with “other”)

Adams	lives	in	California	=: X, input
pers	oth	oth	loc	=: Y, output

X is a vector of words

Y is a vector of class names

Def: Statistical NEA Feature

In statistical NEA, a feature is a function $f(X, i, y) \in R$ that maps

- a word vector X
- a position i in X
- a class name y

to a real value.

Example:

$$f_1(X, i, y) = 1 \text{if } x_i \text{ is CapWord} \wedge y = \text{"Name"}$$
$$= 0 \quad \text{else}$$

The feature returns 1 if it thinks that one particular annotation is right.

We will assume that features return 0 by default.

$$f_2(X, i, y) := 1 \text{ if } x_i \text{ upcased} \wedge y = \text{"pers"}$$
$$f_3(X, i, y) := 1 \text{ if } x_{i-1} \text{ is title} \wedge y = \text{"pers"}$$
$$f_4(X, i, y) := 1 \text{ if } x_{i-1} = \text{"in"} \wedge x_i \text{ upcased} \wedge y = \text{"loc"}$$

Example: Statistical NEA features

$f_1(X, i, y)$:= 1 if x_i upcased $\wedge$ y =“pers”

$f_1(<\text{Arthur, talks}>, 1, pers)$

$f_1(<\text{Arthur, talks}>, 2, pers)$

$f_1(<\text{Arthur, talks}>, 1, loc)$

Example: Statistical NEA features

$f_1(X, i, y)$:= 1 if x_i upcased $\wedge$ y =“pers”

$f_1($<Arthur, talks>$, 1, pers) = 1$

$f_1($<Arthur, talks>$, 2, pers) = 0$

$f_1($<Arthur, talks>$, 1, loc) = 0$

Example: Statistical NEA features

$f_2(X, i, y)$:= 1 if x_{i-1} is title $\wedge$ y =“pers”

$$f_1(<\text{Mr., Arthur}>, 1, pers) = 0$$
$$f_1(<\text{Mr., Arthur}>, 2, pers) = 1$$
$$f_1(<\text{Mr., Arthur}>, 1, loc) = 0$$

Def: Statistical NEA

Given

- a corpus vector $X =< x_1, ..., x_m >$
- a vector of features $F =< f_1, f_2, ..., f_n >$
- a weight vector $W =< w_1, w_2, ..., w_n > \in R^n$

compute class names $Y =< y_1, ..., y_n >$

that maximize $\sum_i \sum_j w_j f_j(X, i, y_i)$.

"Find class names for the words, s.t.
each feature is happy for each word."

Example: Statistical NEA

X =<Dr., Dent>

$f_1(X, i, y)$:= 1 if x_i upcased word $\wedge y$ ="loc"

$f_2(X, i, y)$:= 1 if x_{i-1} is title $\wedge y$ ="pers"

$w_1 = 2, w_2 = 5$

Find $Y = < y_1, y_2 >$

that maximizes $\sum_i \sum_j w_j f_j(X, i, y_i)$

for every feature j

for every position i

Finding this Y is usually done by dynamic programming.

Here, we do it by hand.

Example: Statistical NEA

$X =<\text{Dr., Dent}>$

$f_1(X,i,y) := 1$ if x_i upcased word $\wedge y =$"loc"

$f_2(X,i,y) := 1$ if x_{i-1} is title $\wedge y =$"pers"

$w_1 = 2, w_2 = 5$

Find $Y =< y_1, y_2 >$

that maximizes $\sum_i \sum_j w_j f_j(X,i,y_i)$

for every feature j

for every position i

Try all Y

$Y =< oth, loc >$: $2 \times 0 + 5 \times 0 + 2 \times 1 + 5 \times 0$

$Y =< oth, per >$: $2 \times 0 + 5 \times 0 + 2 \times 0 + 5 \times 1$

Example: Statistical NEA

X =<Dr., Dent>

$f_1(X, i, y)$:= 1 if x_i upcased word $\wedge y$ ="loc"

$f_2(X, i, y)$:= 1 if x_{i-1} is title $\wedge y$ ="pers"

$w_1 = 2, w_2 = 5$

Find $Y =< y_1, y_2 >$

that maximizes $\sum_i \sum_j w_j f_j(X, i, y_i)$

for every feature j

for every position i

Try all Y

$i = 1, x_i =$ Dr.

$Y =< oth, loc >$: $2 \times 0 + 5 \times 0 + 2 \times 1 + 5 \times 0$

$Y =< oth, per >$: $2 \times 0 + 5 \times 0 + 2 \times 0 + 5 \times 1$

Example: Statistical NEA

$X =$<Dr., Dent>

$f_1(X, i, y) :=$ 1 if x_i upcased word $\wedge y =$"loc"

$f_2(X, i, y) :=$ 1 if x_{i-1} is title $\wedge y =$"pers"

$w_1 = 2, w_2 = 5$

Find $Y =< y_1, y_2 >$

that maximizes $\sum_i \sum_j w_j f_j(X, i, y_i)$

for every feature j

for every position i

Try all Y

$i = 1, x_i =$ Dr.

$w_1 \times f_1 \quad w_2 \times f_2$

$Y =< oth, loc >$: $2 \times 0 + 5 \times 0 + 2 \times 1 + 5 \times 0$

$Y =< oth, per >$: $2 \times 0 + 5 \times 0 + 2 \times 0 + 5 \times 1$

Example: Statistical NEA

$X =$ <Dr., Dent>

$f_1(X, i, y) :=$ 1 if x_i upcased word $\wedge y =$"loc"

$f_2(X, i, y) :=$ 1 if x_{i-1} is title $\wedge y =$"pers"

$w_1 = 2, w_2 = 5$

Find $Y =< y_1, y_2 >$

that maximizes $\sum_i \sum_j w_j f_j(X, i, y_i)$

for every feature j

for every position i

Try all Y

$i = 1, x_i =$ Dr. $i = 2, x_i =$ Dent

$w_1 \times f_1 \; w_2 \times f_2$

$Y =< oth, loc >: 2 \times 0 + 5 \times 0 + 2 \times 1 + 5 \times 0$

$Y =< oth, per >: 2 \times 0 + 5 \times 0 + 2 \times 0 + 5 \times 1$

Example: Statistical NEA

X =<Dr., Dent>

$f_1(X,i,y)$:= 1 if x_i upcased word $\wedge y$ ="loc"

$f_2(X,i,y)$:= 1 if x_{i-1} is title $\wedge y$ ="pers"

$w_1 = 2, w_2 = 5$

Find $Y =< y_1, y_2 >$

that maximizes $\sum_i \sum_j w_j f_j(X,i,y_i)$

for every feature j

for every position i

Try all Y

$i = 1, x_i =$ Dr. $i = 2, x_i =$Dent

$w_1 \times f_1 \; w_2 \times f_2 \quad w_1 \times f_1 \, w_2 \times f_2$

$Y =< oth, loc >$: $2 \times 0 + 5 \times 0 + 2 \times 1 + 5 \times 0$

$Y =< oth, per >$: $2 \times 0 + 5 \times 0 + 2 \times 0 + 5 \times 1$

Example: Statistical NEA

X =<Dr., Dent>

$f_1(X, i, y)$:= 1 if x_i upcased word $\wedge y$ ="loc"

$f_2(X, i, y)$:= 1 if x_{i-1} is title $\wedge y$ ="pers"

$w_1 = 2, w_2 = 5$

Find $Y =< y_1, y_2 >$

that maximizes $\sum_i \sum_j w_j f_j(X, i, y_i)$

for every feature j

for every position i

Try all Y

$i = 1, x_i =$ Dr. $\quad i = 2, x_i =$ Dent

$w_1 \times f_1 \;\; w_2 \times f_2 \quad w_1 \times f_1 \;\; w_2 \times f_2$

$Y =< oth, loc >$: $2 \times 0 + 5 \times 0 + 2 \times 1 + 5 \times 0 = 2$

$Y =< oth, per >$: $2 \times 0 + 5 \times 0 + 2 \times 0 + 5 \times 1 = 5$ winner

task>58

Task: Statistical NEA

Given

$X =$<in, London>

$f_1(X, i, y) := 1$ if x_i upcased $\wedge y =$"pers"

$f_2(X, i, y) := 1$ if x_{i-1}="in" $\wedge y =$"loc"

$f_3(X, i, y) := 1$ if $y =$"other"

$w_1 = 2, w_2 = 5, w_3 = 1$

compute one annotation whose value is larger than 3.

Stat. NEA has complex features

Features in statistical NEA can be any functions, like in rule-based NEA.

$f_{42}(X, i, y) = 1$ if $x_i \in L([A - Z]*) \wedge y =$"name"

Features can be real-valued, too:

$f_{43}(X, i, y) = y =$"country" ? editDist(x_i,"UK") : 0

Features can be complete nonsense, too:

$f_{44}(X, i, y) = 1$ if $x_i =$"Arthur" $\wedge y =$"loc"

This will not hurt if $w_{44} = 0$.

Overview

- Named Entity Annotation (NEA)
- NEA by rules
  - Learning NEA rules
- NEA by statistical models
  - Learning statistical NEA models

Learning the weights

Given

- a corpus vector $X =< x_1, ..., x_m >$
- a vector of features $F =< f_1, ..., f_n >$
- a weight vector $W =< w_1, ..., w_n > \in R^n$

compute class names $Y =< y_1, ..., y_m >$

Where do we get the weights from?

How to build a stat. NEA model

1. Define features $F = < f_1, ..., f_n >$

2. Produce a training corpus
   $X = < x_1, ..., x_m >, Y = < y_1, ..., y_m >$

3. Find weights $W = < w_1, ..., w_n > \in R^n$ for the features
   so that statistical NEA annotates the training corpus correctly
   (i.e., bad features will get low weight, good features high weight).

Learning the weights for stat. NEA

We define the vector of features:

$$F(X, i, y) =< f_1(X, i, y), ..., f_n(X, i, y) >$$

... and a probability distribution over Y :

$$Pr(Y|X, W) := \frac{e^{\sum_i W \times F(X,i,y_i)}}{Z_W(X)} \qquad Z_W(X) := \sum_{Y'} e^{\sum_i W \times F(X,i,y'_i)}$$

Probability
of vector Y
given X and W

Pr(Y—X, W) is
proportional
to e^ happiness
of each feature
for each word

To have a value in [0,1]
and to avoid setting all
weights to infinity, we
normalize by dividing by
the sum of the happiness
of ALL other annotations Y'

Learning the weights for stat. NEA

Goal: Find W that maximizes

$$Pr(Y|X,W) := \frac{e^{\sum_i W \times F(X,i,y_i)}}{Z_W(X)}$$

for training corpus (X,Y) .

I.e., find W that maximizes

$$log(Pr(Y|X,W)) = (\sum_i W \times F(X,i,y_i)) - log(Z_W(X))$$

(X,Y) is the given training corpus.

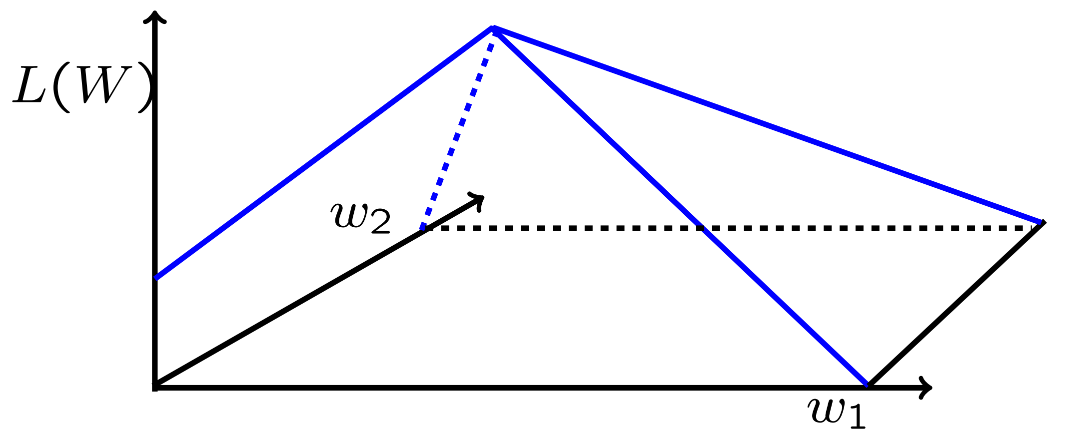
F are given features. Hence, everything

in this formula except W is constant.

Learning the weights for stat. NEA

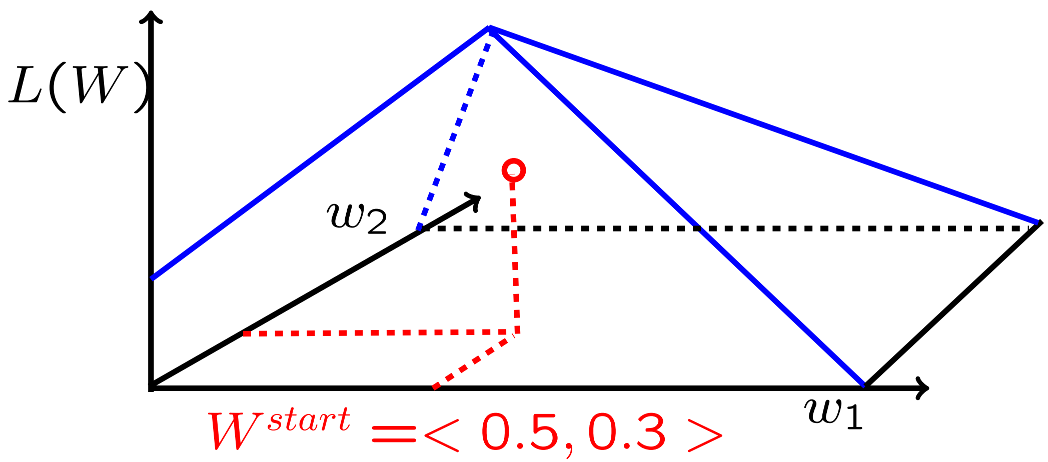
$$L(W) = (\sum_i W \times F(X, i, y_i)) - log(Z_W(X))$$

This function is concave in W:



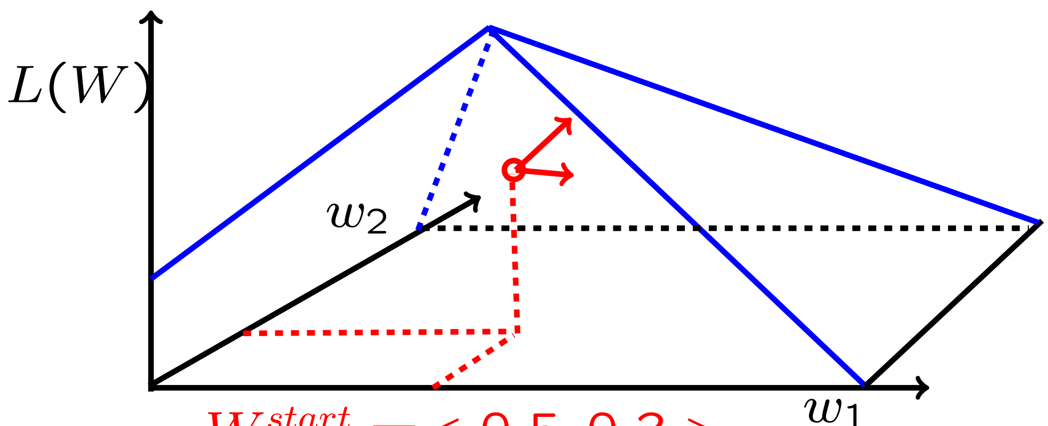
Learning the weights for stat. NEA

1. Start with arbitrary W



Learning the weights for stat. NEA

2. Compute the derivative at W



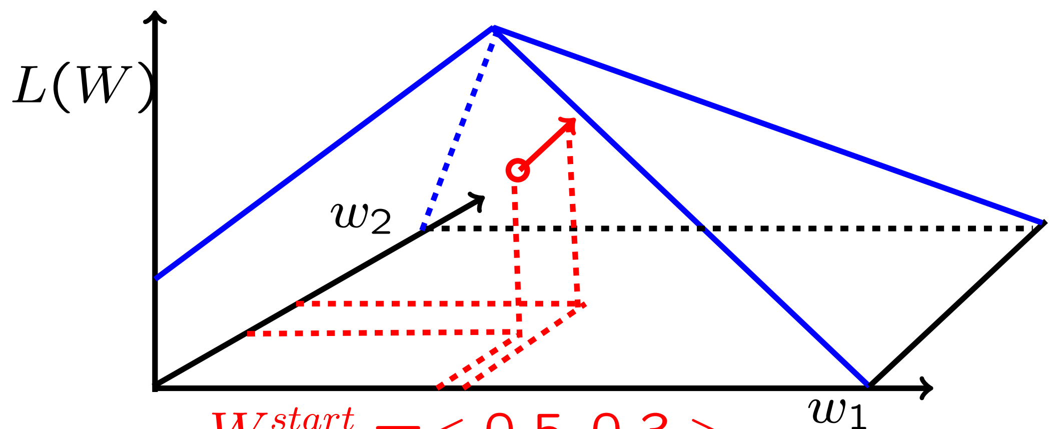
$W^{start} = < 0.5, 0.3 >$

$L'(W^{start}) = < 0.1, 0.9 >$

Go more in direction of w_2 than of w_1

Learning the weights for stat. NEA

3. Move W in the direction of the derivative



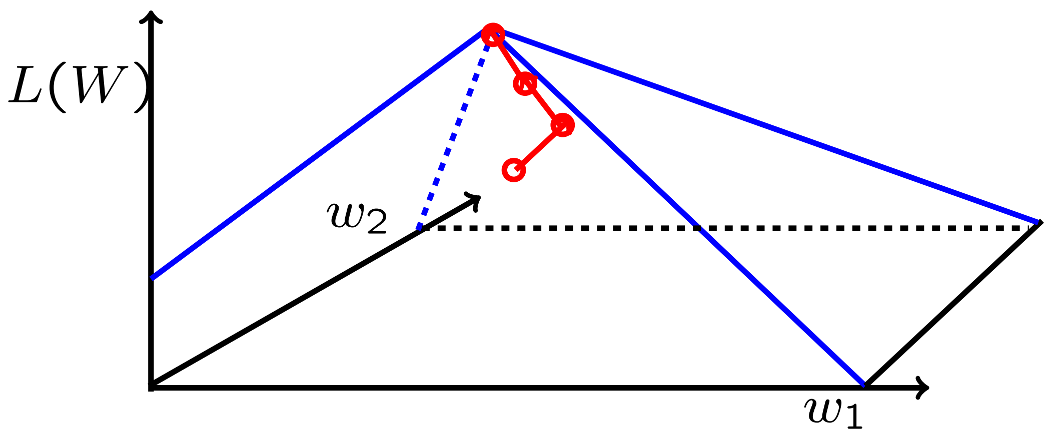
$W^{start} =< 0.5, 0.3 >$

$L'(W^{start}) =< 0.1, 0.9 >$

$$W^{new} = W^{start} + \alpha \times L'(W^{start})$$

Learning the weights for stat. NEA

4. Continue until $L(W)$ is maximal



In the end, the weight vector W will be such that statistical NEA re-produces the manual annotation Y of the training corpus X.

Summary: Statistical NEA

Statistical NEA uses the following notations:

- a corpus $X =< x_1, ..., x_m >$
- class labels $Y =< y_1, ..., y_m >$
- features $F =< f_1, ..., f_n >$
- weights $W =< w_1, ..., w_n >$

Statistical NEA learns the weights W on a manually annotated training corpus (X, Y), as follows:

$$W = argmax_{W'} log(Pr(Y|X, W'))$$

Given a new corpus X', it computes the annotations Y' as

$$Y' = argmax_Y \sum_i W \times F(X', i, y_i)$$

->probabilities

deviation>72

Deviation: Statistical NEA

Task: Find dates such as “May 23rd 2013”

$f_1(X, i, y) = 1$ if x_i is uppercase $\wedge y$ =“date”

$f_2(X, i, y) = 1$ if x_{i-1} is title $\wedge y$ =“date”

Find $Y = argmax_{Y'} \sum_i W \times F(X, i, y'_i)$

Deviation: Statistical NEA

Task: Find dates such as “May 23rd 2013”

$f_1(X, i, y) = 1$ if x_i is uppercase $\wedge y$ =“date”

$f_2(X, i, y) = 1$ if x_{i-1} is title $\wedge y$ =“date”

Find $Y = argmax_{Y'} \sum_i W \times F(X, i, y'_i)$

This will never work!

Machine learning is not magic,

it is never better than its features!

Summary: NEA

NEA finds entity names and
annotates them with predefined classes.

<pers>Arthur</pers> eats at
<org>Milliways</org>

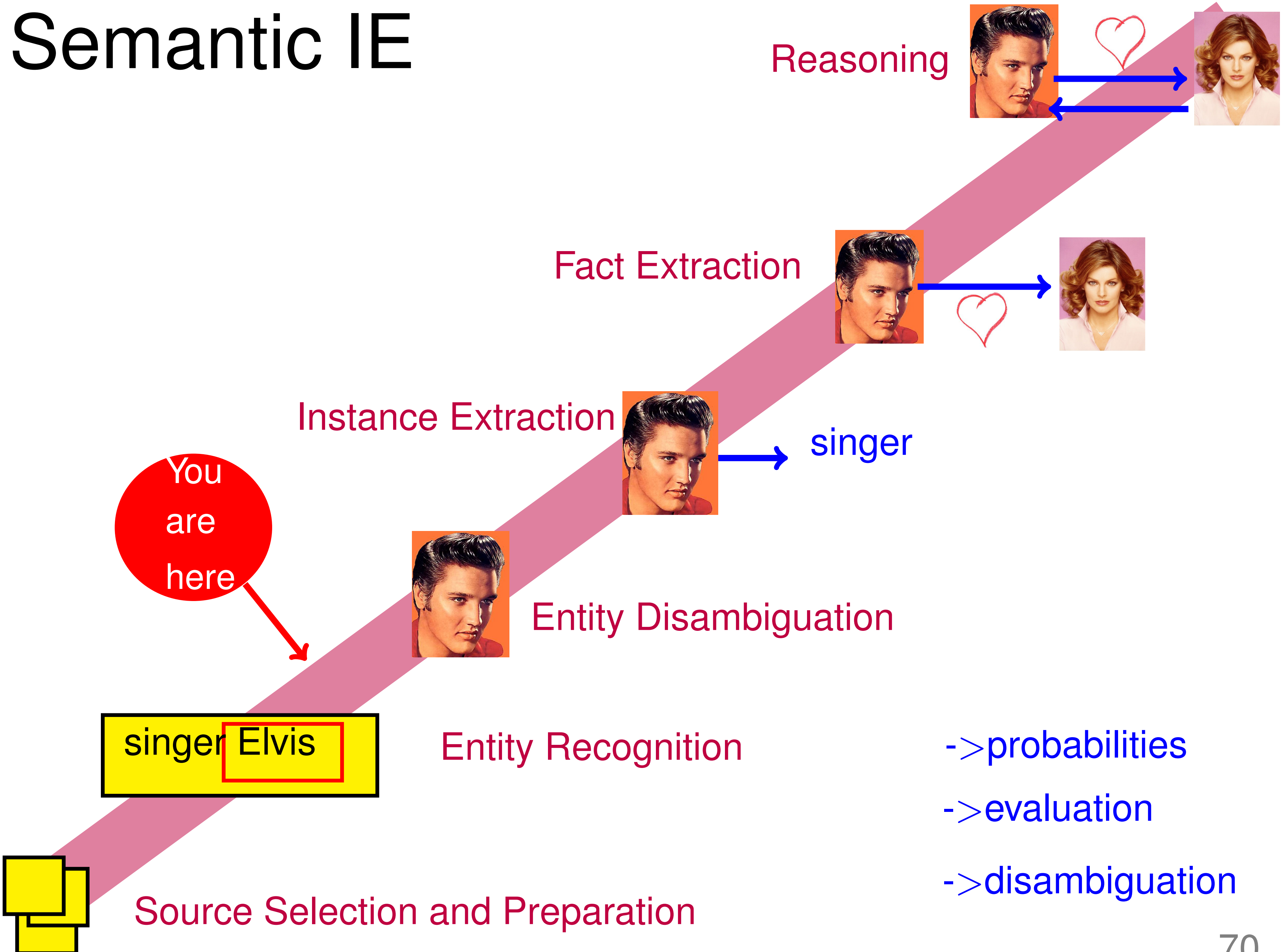
- Rule-based NEA

[CapWord] eats => pers

- Statistical NEA

$argmax_Y \sum_i W \times F(X, i, y_i)$ ->probabilities

Semantic IE


Reasoning
Fact Extraction
Instance Extraction
singer
You
are
here
Entity Disambiguation
singer Elvis
Entity Recognition
->probabilities
->evaluation
->disambiguation
Source Selection and Preparation

References

Sunita Sarawagi: Information Extraction

Diana Maynard: Named Entity Recognition

->evaluation

->disambiguation