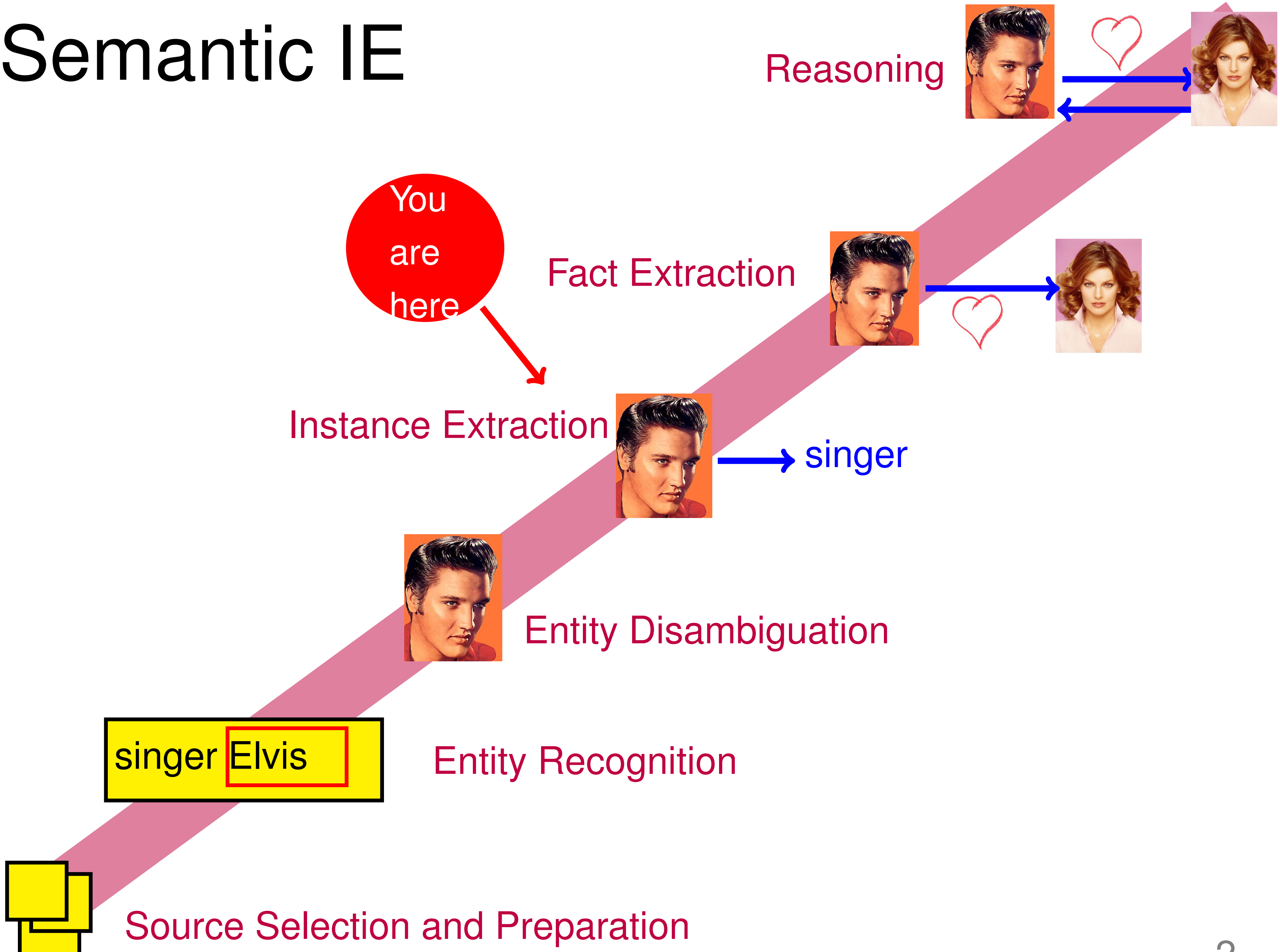
Named Entity Recognition and Classification

Fabian M. Suchanek

Semantic IE


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

Overview

- Named Entity Recognition and Classification (NERC)
- NERC Features
- NERC by rules
- NERC by Machine Learning
- NERC by statistical methods

Def: NE Recognition & Classification

Named Entity Recognition and Classification (NERC) 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.

用一组给定的类中的一个类来注释每个名字。

(This is often called simply “Named Entity Recognition”. We use “Named Entity Recognition and Classification” here to distinguish it from bare NER.)

classes={Person, Location, Organization}

Arthur Dent eats at Milliways.

Def: NE Recognition & Classification

Named Entity Recognition and Classification (NERC) 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 often called simply “Named Entity Recognition”. We use “Named Entity Recognition and Classification” here to distinguish it from bare NER.)

classes={Person, Location, Organization}

Arthur Dent eats at Milliways.

Person

Organization

>examples

Classes

NERC usually focuses the classes person, location, and organization.

But some also extract money, percent, phone number, job title, artefact, brand, product, protein, drug, etc.

ENAMEX

- Person
  - Individual
    - Family name
    - Title
  - Group
- Organization
  - Government
  - Public/private company
  - Religious
  - Non-government
    - Political Party
    - Para military
    - Charitable
    - Association
  - GPE (Geo-political Social Entity)
  - Media
- Location
  - Place
    - District
    - City
    - State
    - Nation
    - Continent
  - Address
  - Water-bodies
  - Landscapes
  - Celestial Bodies
  - Manmade
    - Religious Places
    - Roads/Highways
    - Museum
    - Theme parks/Parks/Gardens
    - Monuments
- Facilities
  - Hospitals
- Institutes
- Library
  - Hotel/Restaurants/Lodges
  - Plant/Factories
  - Police Station/Fire Services
  - Public Comfort Stations
  - Airports
  - Ports
  - Bus-Stations
- Locomotives
- Artifacts
  - Implements
  - Ammunition
  - Paintings
  - Sculptures
  - Cloths
  - Gems & Stones
- Entertainment
  - Dance
  - Music
  - Drama/Cinema
  - Sports
  - Events/Exhibitions/Conferences
- Cuisine's
- Animals
- Plants

[Sobha Lalitha Devi]

>examples

NERC examples

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

in XML

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

in TSV

41 towel OTHER
41 . OTHER
42 Arthur PER
42 Dent PER
42 visits OTHER
42 Milliways ORG

token class

TSV file of
- sentence number
- token
- class

Try this

>examples

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 ...

>examples

NERC 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

- Ambiguity

  May (month, person, or verb?), Washington, etc.

Overview

- Named Entity Recognition and Classification (NERC)
- NERC Features
- NERC by rules
- NERC by Machine Learning
- NERC by statistical methods

Window

token是形成单位的字符序列，例如单词，标点符号，数字等。

A token is a sequence of characters that forms a unit, such as a word, a punctuation symbol, a number, etc.

A window of width δ of a token t in a corpus is the sequence of δ tokens before t, the token t itself, and δ tokens after t.

“You know” said Arthur “I really wish I’d listened to what my mother told me when I was young.” — “Why, what did she tell you?” “I don’t know, I didn’t listen.”

Window of width $\delta = 3$ around “Arthur”:

[know , ”, said, Arthur, “, I, really]

Position -1 Position 0 Position +1

NERC Feature

An NERC feature is a property of a token that could indicate a certain NERC class for the main token of a window.

	[know	,	”,	said,	Arthur,	“,	I,	really]
is stopword	0	0	0	0	0	1	1
matches [A-Z][a-z]+	0	0	0	1	0	0	0
is punctuation	0	1	0	0	1	0	0

Syntactic Features

- 大写字母 capitalized word — Fenchurch
- all upper case word — WSOGMM
- smallcase word — planet
- mixed letters/numbers — HG2G
- number — 42
- special symbol — a|
- space
- punctuation — .,;:?!
- Regular expression — [A-Z][a-z]+

The Stanford NERC system uses string patterns:

Paris —> Xxxx

M2 D&K —> X# X&X

+33 1234 —> +## ####

[Jenny Rose Finkel]

Dictionary Features

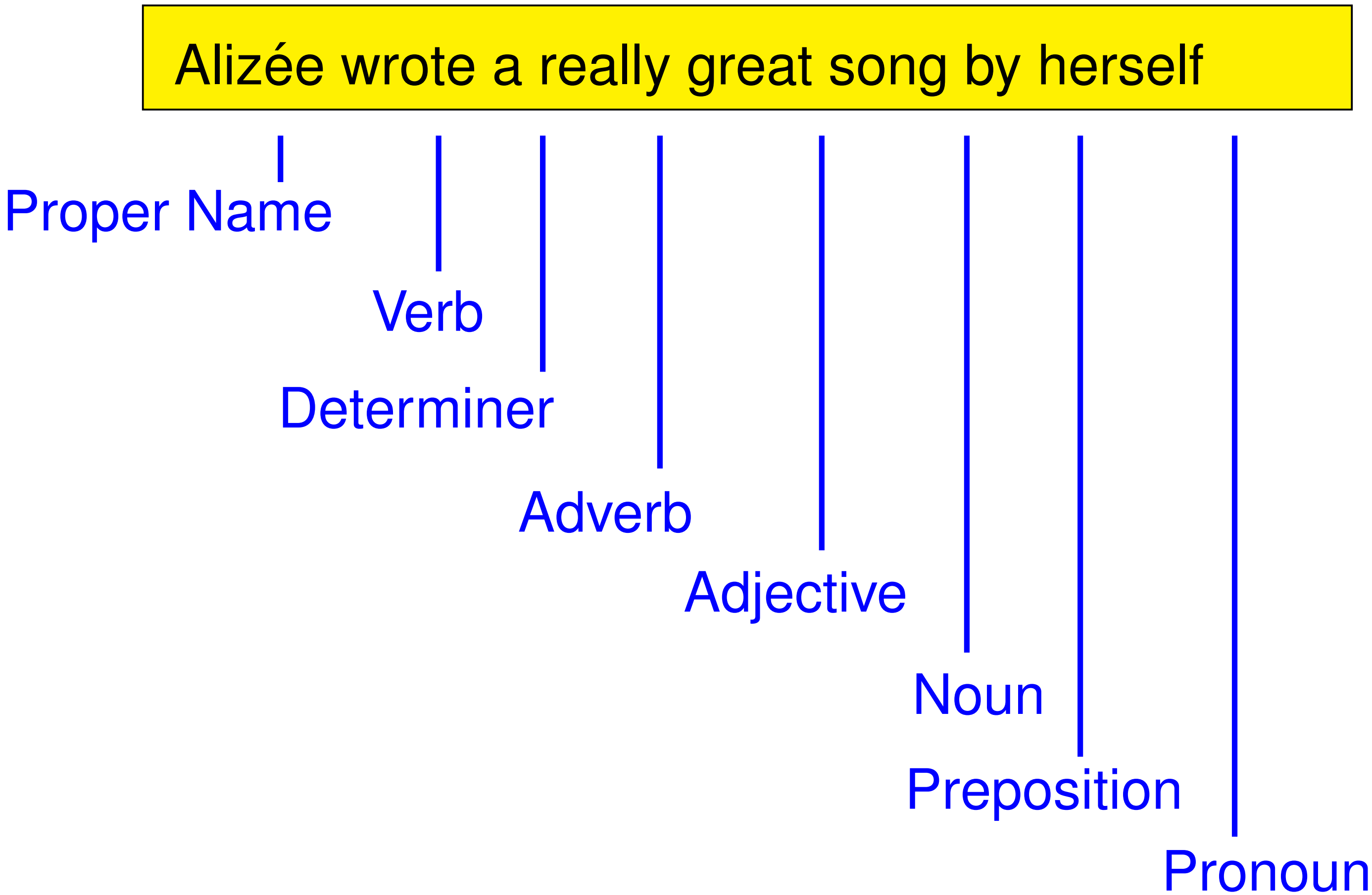
• cities	Vassilian
• countries	UK
• titles	Dr.
• common names	Arthur
• airport codes	CDG
• words that identify a company	Inc, Corp, ...
• common nouns	car, president, ...
• hard-coded words	M2

... if you have a dictionary.

Def: POS

词性（词性）（POS：POS-标签，词类，词汇类，词汇类）是一组具有相同语法作用的词。

A Part-of-Speech (also: POS, POS-tag, word class, lexical class, lexical category) is a set of words with the same grammatical role.



POS Tag Features for NERC

DT	Determiner 限定词
IN	Preposition or subordinating conjunction 介词或从属连词
JJ	Adjective
NN	Noun, singular or mass
NNP	Proper noun, singular 适当的名词，单数
PRP	Personal pronoun
RB	Adverb
SYM	Symbol
VBZ	Verb, 3rd person singular present
...	

[Penn Treebank symbols]

Morphological features

形态特征

- word endings -ish, -ist, ...
- word contains an n-gram Par, ari, ris
- word contains n-grams at boundaries #Par, ari, ris#

直觉：通常，词的形态给出了它的类型的暗示

Intuition: quite often, the morphology of the word gives a hint about its type. Examples:

London Bank —> ORG

Cotramoxazole —> DRUG

Overview

- Named Entity Recognition and Classification (NERC)
- NERC Features
- NERC by rules
- NERC by Machine Learning
- NERC by statistical methods

Def: NERC by rules

NERC by rules uses rules of the form

$$f_1^1, ...f_k^1 [f_1^2, ...f_l^2] f_1^3, ...f_m^3 => c$$

...where $f_1^1 ... f_m^3$ are features and c is a class.
$f_1^2, ...f_l^2$ are the designated features.

If a window of tokens matches $f_1^1 ... f_m^3$, we annotate the tokens of the designated features with c.

"in" [([A-Z][a-z]+)] "City" => Location

She works in London City each day.

Location

Examples for NERC rules

指定的功能

Designated features

Dictionary:Title "." [CapWord{2}] => Person

Features

[CapWord (Street—Av—Road)] => Location

"a pub in" [CapWord] => Location

"based in" [CapWord] => Location

"to the" Compass "of" [CapWord] => Location

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

NERC examples from GATE

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

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

Task: NERC patterns

Design NERC 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: NERC 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 NERC rules

If two NERC 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” => illness疾病

He lives on Ursa Minor IV.

如果两个NERC规则匹配重叠的字符串，我们必须决定应用哪一个。 可能的策略：
•仅使用具有较长匹配的规则进行注释
•手动定义优先顺序

cascade>28

Def: Cascaded NERC

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

级联的NERC将NERC应用于之前NERC运行注释的语料库。

Main Street 42, West City

First NERC run

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

Second NERC run

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

Cascaded NERC rule:

[Street City] => Adr

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

>task

Task: Cascaded NERC

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

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

Possible Solution: Cascaded NERC

First run:

Dictionary:AcademicTitle => Title

Dictionary:FrenchTitle => Title

Dictionary:EnglishTitle => Title

CapWord-CapWord => Name

CapWord => Name

Second run:

Title Name Name => Person

Matching NERC rules

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

One possibility is to hard-code the rule:

```
if(window.getWordAt(-1)==”in”) return(LOCATION);
...
```

Another possibility is to compile the rule to a regex:

```
Dict:Title [CapWord] => Person

(Dr—Prof—Mr—Ms) [A-Z][a-z]+ => Person
```

Learning NERC Rules

NERC rules are often 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

<pers>Arthur</pers> says “Hello”

1. Find a NERC rule for each annotation

[Arthur] ”says “Hello”” => pers

Example: Rule learning

0. Start with annotated training corpus

1. Find a NERC rule for each annotation

<pers>Arthur</pers> says “Hello”

[Arthur] ”says “Hello”” => pers

[Ford] ”says “Hello”” => pers

Example: Rule learning

0. Start with annotated training corpus

1. Find a NERC rule for each annotation

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

<pers>Arthur</pers> says “Hello”

[Arthur] ”says “Hello”” => pers

[Ford] ”says “Hello”” => pers

Generalize

[CapWord] ”says “Hello”” => pers

Example: Rule learning

0. Start with annotated training corpus

<pers>Arthur</pers> says “Hello”

1. Find a NERC 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 NERC 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

1. Find a NERC 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

[CapWord] (says—yells—screams)=>pe

Example: Rule learning

0. Start with annotated training corpus

<pers>Arthur</pers> says “Hello”

1. Find a NERC 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~~

4. Remove redundant rules

[CapWord] (says—yells—screams)=>pe

5. Repeat

NERC rule learning is not easy

Then [Ford] ”says ’Hello’” => pers

And [Arthur] ”yells ’Bye’” => pers

NERC rule learning is not easy

Then [Ford] ”says ’Hello’” => pers

And [Arthur] ”yells ’Bye’” => pers

Conj

Cap

Word

Drop

NERC rule learning is not easy

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

NERC 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 NERC rule learning

Learn rules that

- cover all training examples (high recall)
- don’t cover non-annotated strings (high precision)
- are not too specific/numerous

  (we do not want 1 rule for each annotation)

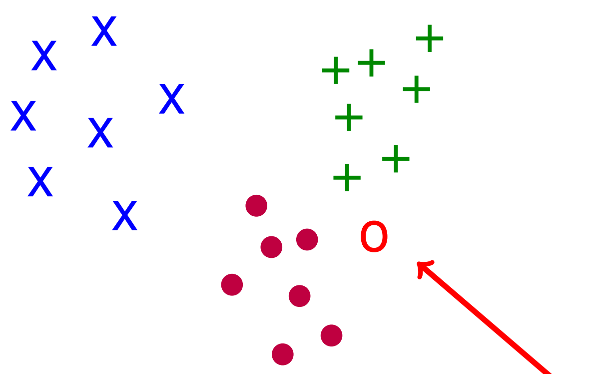
Overview

- Named Entity Recognition and Classification (NERC)
- NERC Features
- NERC by rules
- $\boxed{\text{NERC by Machine Learning}}$
- NERC by statistical methods

NERC by Machine Learning

NERC can be seen as a classification task:

- given training examples (a corpus tagged with classes)

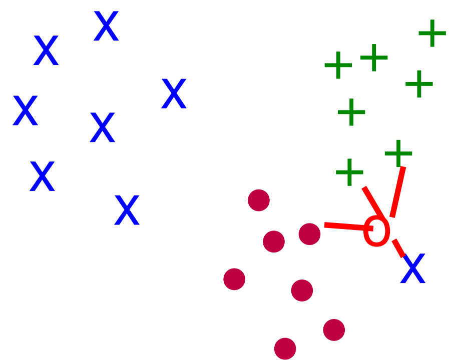


- determine the class of an untagged word

Any classification algorithm can be used: Hidden Markov Models, k Nearest Neighbors, Decision Trees, SVMs, ...

kNN

kNN (k nearest neighbors) is a simple classification algorithm that assigns the class that dominates among the k nearest neighbors of the input element.



The class + dominates among the $k = 4$ nearest neighbors of o

=> classify o as +

k is a constant that is fixed a priori. It serves to make the algorithm more robust to noise. To avoid ties, k is chosen odd.

k是一个先验固定的常数。 它的作用是使算法对噪声更加鲁棒。 为了避免关系，k被选择为奇数。

kNN（和其他分类算法）需要示例上的距离函数。

kNN (and other classification algorithms) require a distance function on the examples.

Naïve distance function

Represent each example window as a vector

¡$f_1^{-1}, f_2^{-1}, \ldots, f_n^{-1}, f_1^{0}, \ldots, f_1^{+1}, \ldots$¿

$f_i^j = 1$ if Feature i applies to the token at position j relative to the main token

Everyone loves <per>Fenchurch</per> because...

$f_1 =$ is upper case

$f_2 =$ is stopword

Naïve distance function

Represent each example window as a vector

$$¡f_1^{-1}, f_2^{-1}, ..., f_n^{-1}, f_1^0, ..., f_1^{+1}, ...¿$$

$f_i^j = 1$ if Feature i applies to the token at position j relative to the main token

Everyone loves <per>Fenchurch</per> because...

f_1^{-1} f_2^{-1} f_1^0 f_2^0 f_1^{+1} f_2^{+1}

$$¡0, 0, 1, 0, 0, 1¿$$

$f_1 =$ is upper case

$f_2 =$ is stopword

Naïve distance function

Represent each example window as a vector

¡$f_1^{-1}, f_2^{-1}, ..., f_n^{-1}, f_1^{0}, ..., f_1^{+1}, ..$¿

$f_i^j = 1$ if Feature i applies to the token at position j relative to the main token

Everyone loves <per>Fenchurch</per> because...

f_1^{-1} f_2^{-1} f_1^{0} f_2^{0} f_1^{+1} f_2^{+1}

¡$0, 0, 1, 0, 0, 1$¿

$f_1 =$ is upper case

$f_2 =$ is stopword

Represent the input also as a (potentially weighted) feature vector:

Nobody loves <?>Arthur</?> because...

¡$0, 0, 2, 0, 0, 1$¿

Naïve distance function

Represent each example window as a vector

¡$f_1^{-1}, f_2^{-1}, ..., f_n^{-1}, f_1^0, ..., f_1^{+1}, ...$¿

$f_i^j = 1$ if Feature i applies to the token at position j relative to the main token

Everyone loves <per>Fenchurch</per> because...

f_1^{-1} f_2^{-1} f_1^0 f_2^0 f_1^{+1} f_2^{+1}

¡$0, 0, 1, 0, 0, 1$¿

$f_1 =$ is upper case

$f_2 =$ is stopword

Represent the input also as a (potentially weighted) feature vector:

Nobody loves <?>Arthur</?> because...

¡$0, 0, 2, 0, 0, 1$¿

Then use the Euclidian distance.

Overview

- Named Entity Recognition and Classification (NERC)
- NERC Features
- NERC by rules
- NERC by Machine Learning
- NERC by statistical methods

Def: Statistical NERC corpus

A corpus for statistical NERC is a vector of tokens.

The output is a vector of class names.

(Tokens 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 tokens

Y is a vector of class names

Def: Statistical NERC Feature

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

- a token 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.

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

->Summary

Example: Statistical NERC features

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

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

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

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

Example: Statistical NERC features

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

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

Example: Statistical NERC 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 NERC

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.”

$>$examples

Example: Statistical NERC

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

$f_1(X, i, y) :=$ 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

Finding this Y is usually done by dynamic programming.

Here, we do it by hand.

Example: Statistical NERC

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

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

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

$w_1 = 2, w_2 = 5$

(i= 1)
2 * f1(x,1,Other) + 5 * f2(X,1,Other) +
(i= 2)
2 * f1(X,2,loc) + 5 * f2(x,2,loc)
= 2 * 0 + 5 * 0 + 2 * 1 + 5 * 0

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 NERC

$X =$<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 NERC

$X =$<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 NERC

$X = <$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

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

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

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

Example: Statistical NERC

$X =$<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

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

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

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

Example: Statistical NERC

$X =$<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

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

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

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

>task

Task: Statistical NERC

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. NERC has complex features

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

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

Features can be real-valued, too:

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

Features can be complete nonsense, too:

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

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

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. NERC 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 NERC annotates the training corpus correctly
   (i.e., bad features will get low weight, good features high weight).

>details

Learning the weights for stat. NERC

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. NERC

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.

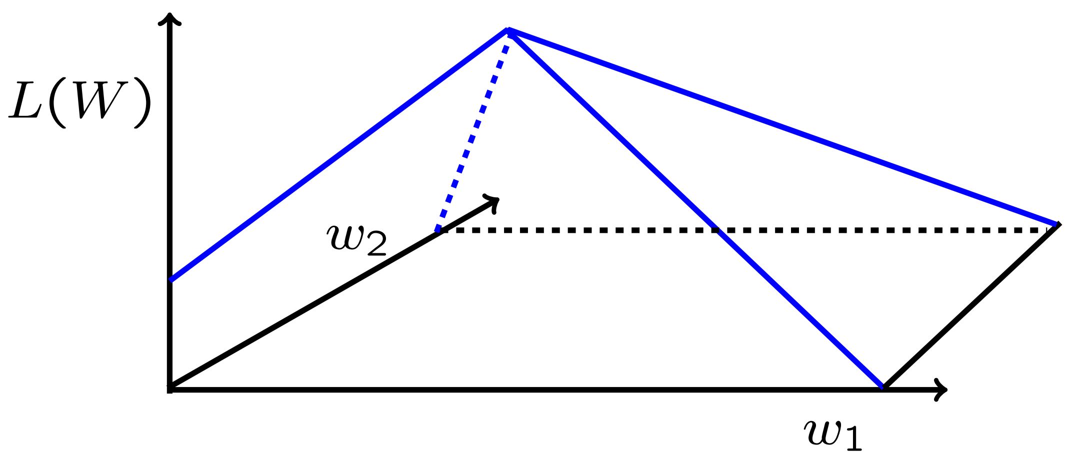
F are given features. Hence, everything

in this formula except W is constant.

Learning the weights for stat. NERC

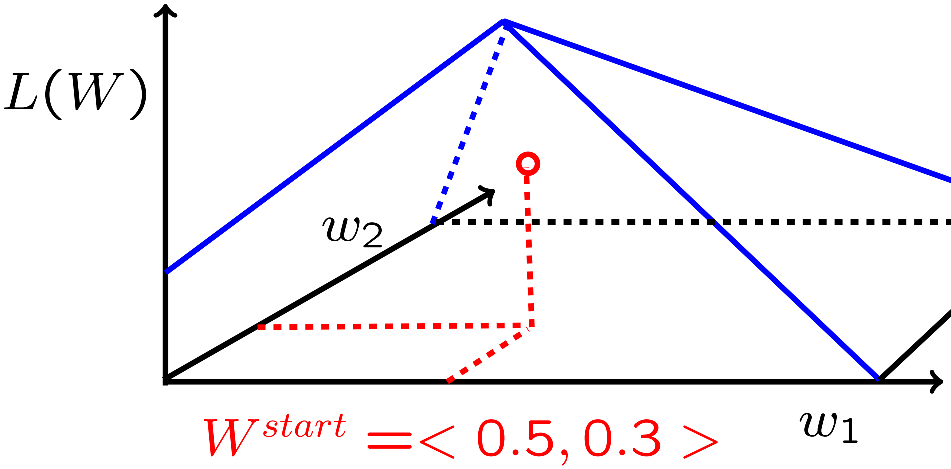
$$L(W) = (\sum_i W \times F(X, i, y_i)) - log(Z_W(X))$$

This function is concave in W:



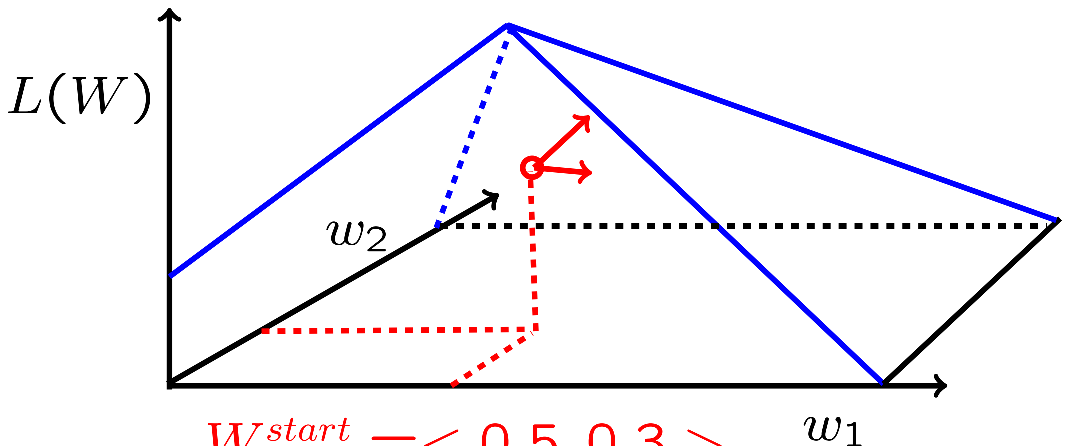
Learning the weights for stat. NERC

1. Start with arbitrary W



Learning the weights for stat. NERC

2. Compute the derivative at W



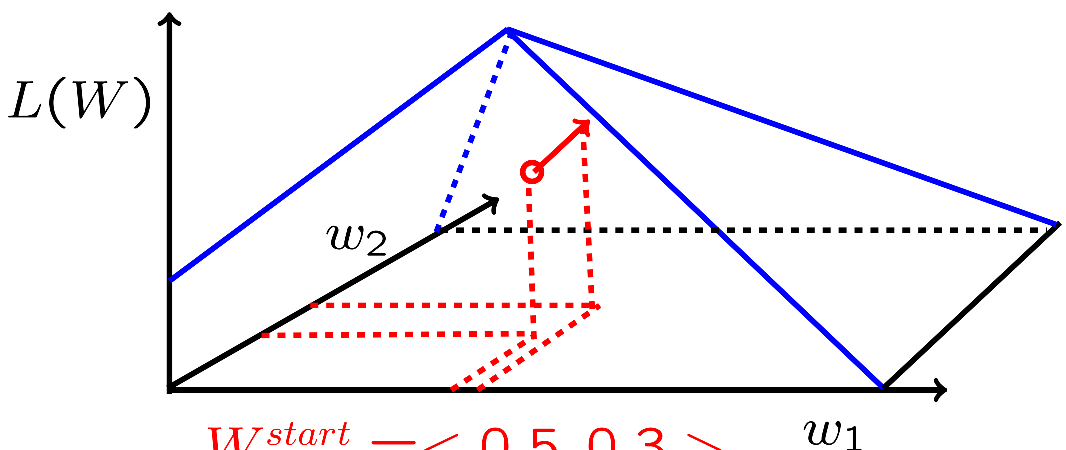
$W^{start} = < 0.5, 0.3 >$

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

Go more in direction of w_2 than of w_1

Learning the weights for stat. NERC

3. Move W in the direction of the derivative



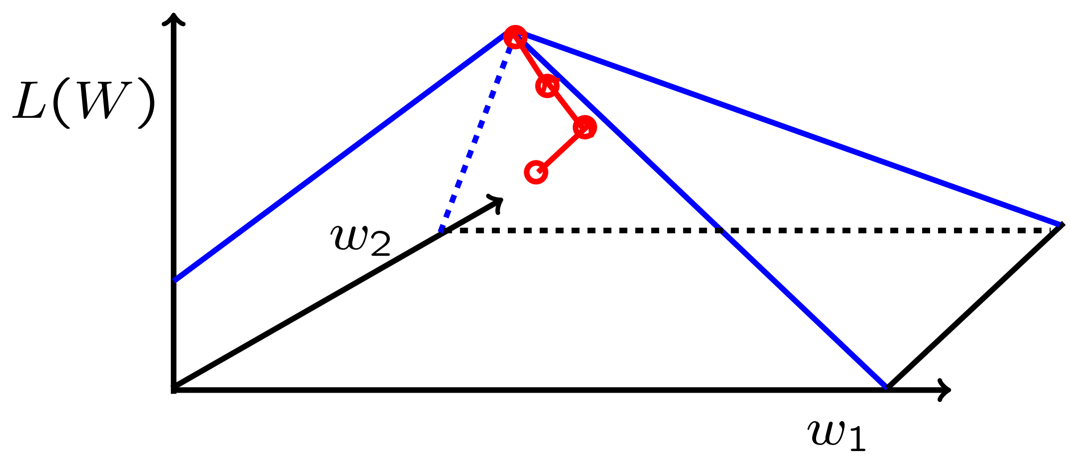
$$W^{start} = < 0.5, 0.3 >$$

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

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

Learning the weights for stat. NERC

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



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

Summary: Statistical NERC

Statistical NERC 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 NERC 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 NERC

Task: Find dates such as “May 23rd 2017”

$f_1(X, i, y) = 1$ 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 NERC

Task: Find dates such as "May 23rd 2017"

$f_1(X, i, y) = 1$ 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: NERC

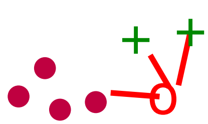
NERC (named entity recognition and classification) finds entity names and annotates them with predefined classes.

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

- Rule-based NERC

  [CapWord] eats => pers

- NERC by Machine Learning

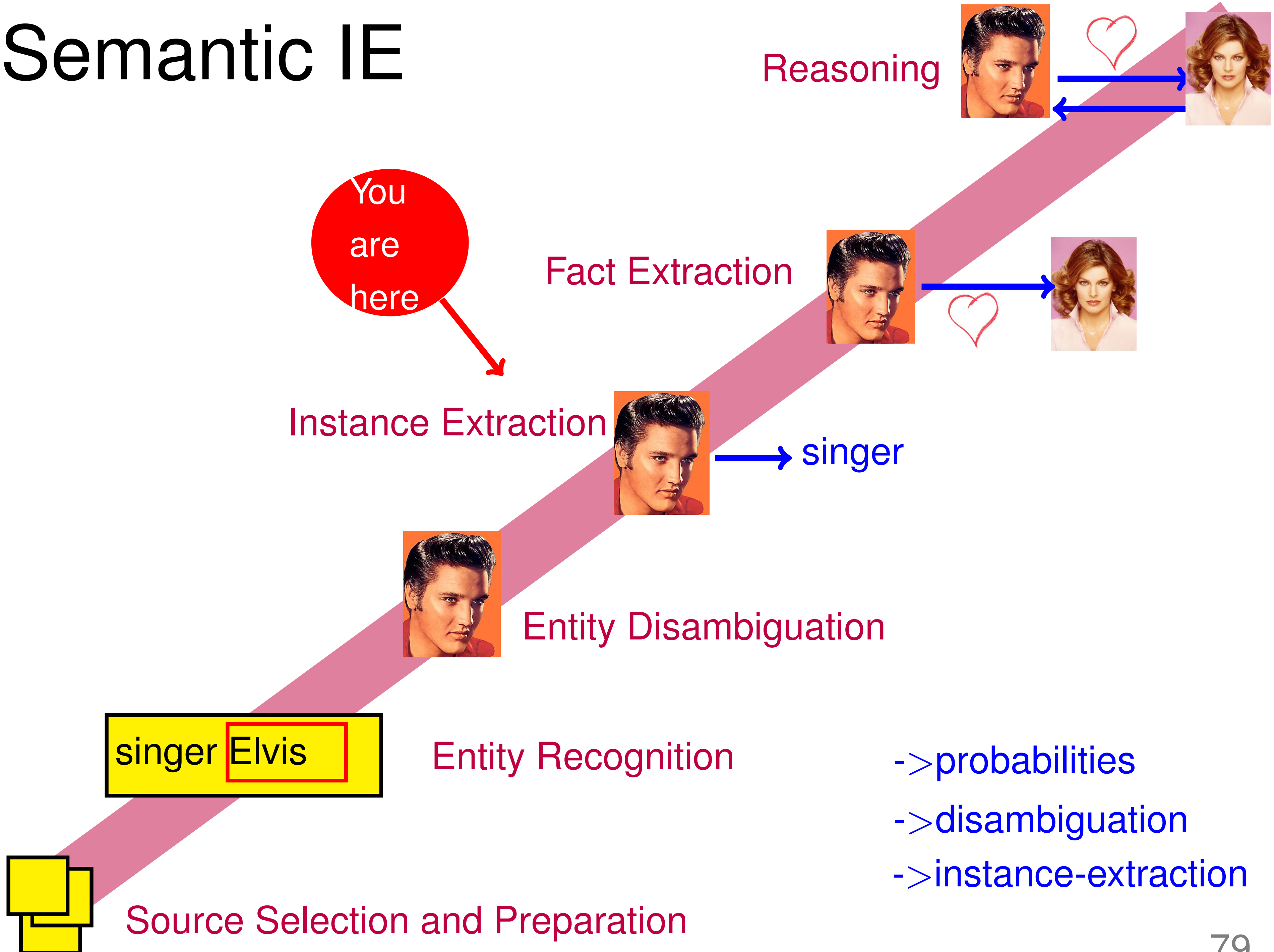


- Statistical NERC

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

  ->probabilities

Semantic IE



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

References

Sunita Sarawagi: Information Extraction

Diana Maynard: Named Entity Recognition

->probabilities

->disambiguation

->instance-extraction