

Information extraction

4. Entity Typing

Simon Razniewski
Winter semester 2019/20

Announcements

- Plagiarism
 - Sharing solutions not permitted
 - Don't be naive
 - Self report
- Campus lecture series
- EMNLP: New end-to-end benchmark for IE:
<https://github.com/diffbot/knowledge-net>

Entity typing: Motivation

- Einstein: Scientist, Physicist, Nobel prize winner
- Dudweiler: Village, location, municipality
- RCH_2OH : chemical formula, psychoactive substance
- Entities are the basic block of KBs (cf. lecture 2)
- Types often most salient info about entities
- Types help to
 - Organize entities
 - Power search applications
 - Direct and constrain fact extraction
 - ...

Types and fact extraction

In the interwar period, Einstein worked on the unified theory.

In Berlin, Einstein worked on the unified theory.

time(workedOn(Einstein, Unified theory), Interwar period)

place(workedOn(Einstein, Unified theory), Berlin)

After three years in Westmore College, Mary moved to Eastless Academy.

After three years in Software Corp., Mary moved to Hardware Inc.

After three years in Corporate Finance, Mary moved to Controlling.

-> studiedAt? workedFor? fieldOfWork?

Outline

1. Named-entity recognition and classification
 1. Problem
 2. Features
 3. Rule-based
 4. Distance-based
2. Extractive typing
3. Use cases
 - spaCy
 - WebIsALOD
 - Entyfi

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

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/Garden
 - 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]

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

TSV file of

- sentence number
- token
- class

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 ...

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 took the bus from Saarland University

Saarland University has 25 Bachelor programs

- Ambiguity

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

Diana Maynard: Named Entity Recognition

Outline

1. Named-entity recognition and classification

1. Problem
2. Features
3. Rule-based
4. Distance-based

2. Extractive typing

3. Use cases

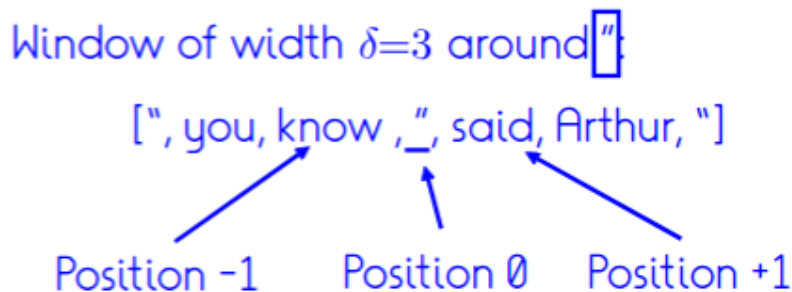
- spaCy
- WebIsALOD
- Entyfi

Window

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

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 "said":

[you, know, ", said, Arthur, ", I]

Position -1

Position 0

Position +1

Window

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, <u>Arthur</u> , ", 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

- capitalized word
- all upper case word
- smallcase word
- mixed letters/numbers
- number
- special symbol
- punctuation
- Regular expression

Fenchurch

WSOGMM

planet

HG2G

42

⌘

.,,:?!

[A-Z][a-z]+

The Stanford NERC system uses string patterns:

Paris → Xxxx

M2 D&K → X# X&X

+33 1234 → +## ####

Dictionary Features

- cities
- countries
- titles
- common names
- airport codes
- words that identify a company
- common nouns
- hard-coded words

Vassilian

UK

Dr.

Arthur

CDG

Inc, Corp, ...

car, president, ...

M2

... if you have a dictionary.

The dictionary can be implemented, e.g, as a trie.

Michael Loster, Zhe Zuo, Felix Naumann, Oliver Maspfuhl, Dirk Thomas:
"Improving Company Recognition from Unstructured Text by using Dictionaries"
EDBT 2017

Def: POS

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
IN	Preposition or subordinating conjunction
JJ	Adjective
NN	Noun, singular or mass
NNP	Proper noun, singular
PRP	Personal pronoun
RB	Adverb
SYM	Symbol
VBZ	Verb, 3rd person singular present
...	

[Penn Treebank symbols]

Morphological features

- word endings -ish, -ist, ...
- word contains an n-gram Par, ari, ris

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

Dudweiler, Landsweiler, Wellesweiler -> LOC

Cotramoxazole -> DRUG

Embedding features

- Eats, munches, devours, tastes have quite related meanings?

→ (Pretrained) word embeddings can capture similarity without requiring explicit synonymy listings like on WordNet

- Word2vec
- GloVe
- (BERT)
- ...

Outline

1. Named-entity recognition and classification

1. Problem
2. Features
3. Rule-based
4. Distance-based

2. Extractive typing

3. Use cases

- spaCy
- WebIsALOD
- Entyfi

Def: NERC by rules

NERC by rules uses rules of the form

$$f_1^1, \dots, f_k^1 [f_1^2, \dots, f_l^2] f_1^3, \dots, f_m^3 \Rightarrow c$$

...where $f_1^1 \dots f_m^3$ are features and c is a class.

$f_1^2, \dots, f_l^2$ are the **designated features**.

If a window of tokens matches $f_1^1 \dots f_m^3$, 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

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

Designated features

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

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 → Organiza-
tion=:orgName

Rule: OrgDept

Priority: 25

// Matches "Department of Pure Mathematics and Physics"

({Token.string = "Department"} {Token.string = "of"} {Orthography type = up-
perInitial}+ ({Token.string = "and"} {Orthography type = upperInitial}+)?) →
Organization

Sunita Sarawagi: Information Ext

<https://gate.ac.uk/>

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|alan) Adj "planet"

- CapWord: A word that starts with a capital letter.
- "is", "planet": plain strings
- (the|alan): 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

Def: Cascaded NERC

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

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>

[Street City] => Adr

Cascaded NERC rule:

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

...

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

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.

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`~~
`[CapWord] (says|yells|screams) => pers`

4. Remove redundant rules

5. Repeat

NERC rule learning is not easy

Then [Ford] "says "Hello"" => pers

And [Arthur] "yells "Bye"" => pers

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)

→ Impossible trinity (trilemma)

General problem of inductive rule learning/association rule mining

Forward reference: Lecture 8

Outline

1. Named-entity recognition and classification

1. Problem
2. Features
3. Rule-based
4. Distance-based

2. Extractive typing

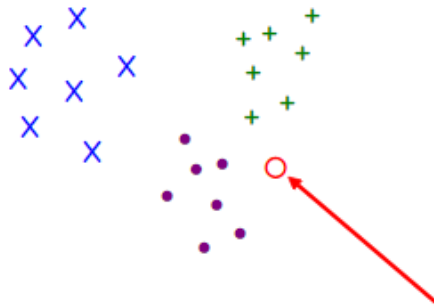
3. Use cases

- spaCy
- WebIsALOD
- Entyfi

NERC by distance

NERC can be seen as a classification task:

- given training examples (a corpus tagged with classes)



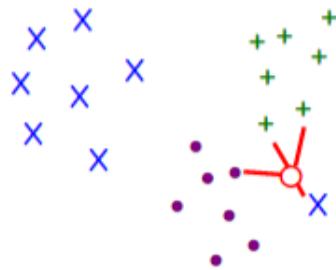
- determine the class of an untagged word

Any classification algorithm can be used:

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**
 $\Rightarrow$ 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.

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

Naive distance function

Represent each example window as a vector

$$\langle f_1^{-1}, f_2^{-1}, \dots, f_n^{-1}, f_1^0, \dots, f_1^{+1} \rangle$$

$f_i^j = 1$ 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

Distant supervision from Wiki[pediala]

- Training data creation costly
- Wiki markup provides disambiguation from surface forms to Wiki entities
 - "The [Max Planck Society | MPG] was founded in 1911 as the Kaiser Wilhelm Society for basic research."
 - "[Monopotassium glutamate | MPG] is used to enhance the taste of ..."
- Wiki categories provide type-like information about entities
 - Max Planck Society → Scientific organization, Research units, ...
 - Monopotassium glutamate → Flavor enhancers, Potassium compounds, E-number additives, ...
- Idea: Train a supervised classifier that predicts categories from mention context
 - "The MPG was founded in 1911 as the Kaiser Wilhelm Society for basic research." → Scientific organization, Research units
 - "MPG is used to enhance the taste of..." → Flavor enhancers, Potassium compounds, E-number additives

Summary: NERC

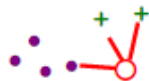
NERC (named entity recognition and classification) finds entity names and annotates them with predefined classes.

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

- Rule-based NERC

[CapWord] eats => pers

- NERC by Machine Learning



Outline

1. Named-entity recognition and classification
 1. Problem
 2. Features
 3. Rule-based
 4. Distance-based
2. Extractive typing
3. Use cases
 - spaCy
 - WebIsALOD
 - Entyfi

Limitations of NERC

- Classification **limited by target set**
 - Traditionally 5-20
 - Lately up to 10k types
 - Nevertheless inherently limited by what was seen
- Single mentions often provide **limited context**
 - Extraction may consider larger text corpus

Example: IsA Extraction

In the Simpson episode "HOMR", Doctor Monson discovers a crayon in Homer's brain and removes it. His IQ goes up from 55 to 105, but he feels uncomfortable and wants it back. Moe, who is not only a bartender but also an unlicensed physician, puts the crayon back, returning Homer to the idiot.

→ Without need for training, types are often mentioned explicitly

Def: IsA

Observation: The relations “subclass” and “type” are expressed very similarly in natural language:

Elvis is a singer.

A dog is an animal.

`type(Elvis, singer)`

`subclassof(dog, animal)`

For now, let us ignore the distinction between the two.

IsA is the relation that holds between x and y
if x is an instance of y , or x is a subclass of y .

`is-a(Elvis, singer)`

`is-a(dog, animal)`

Def: Hearst Patterns

A **Hearst pattern** is a simple textual pattern that indicates an IsA fact that is mentioned implicitly.

"Y such as X"

...idiots such as Homer... → is-a(Homer, idiot)

Def: Hearst Patterns

A **Hearst pattern** is a simple textual pattern that indicates an IsA fact that is mentioned implicitly.

"Y such as X"

...idiots such as Homer... → is-a(Homer, idiot)

...many activists, such as Lisa...

is-a(Lisa, activist)

...some animals, such as dogs...

is-a(dog, animal)

...some scientists, such as computer scientists...

is-a(computer, scientist) ?

...some plants, such as nuclear power plants....

is-a(nuc.Pow.Plants, plants) ?

Hearst patterns need

- NER
- disambiguation
- plural removal

Def: Classical Hearst Patterns

The classical Hearst Patterns are

Y such as X+

such Y as X+

X+ and other Y

Y including X+

Y, especially X+

...where X+ is a list of

names of the form

" $X_1, \dots, X_{n-1}$ (and/or) X_n ".

(In the original paper, the X_i are noun phrases)

These imply $\text{is-a}(X_i, Y)$.

(assuming that the words are noun phrases and disambiguated)

Task: Classical Hearst Patterns

Apply

1. Y such as X+
2. such Y as X+
3. X+ and other Y
4. Y including X+
5. Y, especially X+

I lived in such countries as Germany, France, and Bavaria.

Trump is a candidate for the Nobel Peace Prize, together with Kim-Jon-Un and other world-class-leaders.

I love people that are not genies, especially Homer.

Example: Hearst on the Web

"cities such as"

<https://www.google.com/search?client=firefox-b-d&q=%22cities+such+as%22>

<https://en.wikipedia.org/w/index.php?search=%22cities+such+as%22&title=Special%3FSearch&go=Go&ns0=1>

Example: Hearst in the NELL project

NELL: "Robin"

categories

- **bird**(98.4%)
 - CPL @722 (87.1%) on 06-apr-2013 ["_ sized birds" "_ was singing from" "small birds , such as" "carolling of" "Birds such as" "_ " singing outside" "_ twittered from" "_ flew away as" "plumage of" "nesting habits of" "_ 's feathers" "owl , black" "_ "vert between" "_ "various birds including" "_ " 's feather" "_ 's nests" "_ is a small bird" "bird called" "_ " builds a nest" "_ was the first bird" "_ breeding habitat" "_ is the state bird" "interesting birds like" "_ " swoop out of" "common birds like" "_ " nesting success" "_ 's eye view" "smaller birds such as" "_ " flew right in" "DNS round" "_ "red breasted" "_ " is the only bird" "_ returned yesterday" "_ 's beak" "birds including" "_ "birds , like" "_ "tail feathers of" "_ "favorite bird is" "_ " building a nest" "_ foraging amongst" "Birds , including" "_ " does not make a spring" "birds , including" "_ " had made a nest" "_ 's nest high" "_ 's egg" "_ has a brown back" "_ 's egg blue" "birds include" "_ "migratory birds such as" "_ " had built a nest" "birds , except" "_ "birds , such as" "_ "songbirds like" "_ "flock of American" "_ "bird , whether" "_ "bird known as" "_ " carrying twigs" "nest sites for" "_ " 's plumage" "_ s nest" "unusual sighting of" "_ " 's egg blue color" "birds like" "_ "colony of little" "_ "Island subspecies of" "_ "birds such as" "_ " is a thrush" "_ has built a nest" "_ s nest bed" "_ has nested at" "_ is a denizen" "larger birds such as" "_ " 's wings" "_ kept diving" "ground birds such as" "_ "garden birds such as" "_ " 's nest" "many birds such as" "_ "larger birds like" "_ " is also the state bird" "bird such as" "_ "arrival was heralded by" "_ "State bird is" "_ "Red breasted" "_ " was banded in" "first bird was" "_ " 's chirp" "wing feathers of" "_ " flying overhead on" "_ has made a nest" "_ has a yellow beak" "other birds , such as" "_ "wild birds , such as" "_ " is a songbird" "_ eats insects" "small birds such as" "_ "such birds as" "_ " built a nest" "cackle of" "_ "songbirds include" "_ " is the national bird" "state bird is" "_ "Common birds include" "_ "winter birds such as" "_ " is a migratory species"] using robin

Problems with Hearst Patterns

Hearst Patterns won't extract the right is-a facts from

- ...domestic animals other than dogs such as cats ...
- ...companies such as IBM, Nokia, Proctor and Gamble ...
- ...classic movies such as Gone with the Wind ...
- ...people in Europe, Russia, Brazil, China, and other countries ...

Determining the right super-concept

- ...domestic animals other than dogs such as cats ...
- 

1. Extract all possible super-concepts and all possible sub-concepts

is-a(cat, dog) ?

is-a(cat, domestic animal) ?

2. Choose the most likely one, given what we have seen before

We have seen is-a(cat, animal) more often than is-a(cat, dog)

=> is-a(cat, domestic animal)

Determining the right sub-concept

- ...people in Europe, Russia, Brazil, China, and other countries ...
- 
- The diagram illustrates the process of determining the right sub-concept. It shows a list of words: 'Europe', 'Russia', 'Brazil', 'China', and 'countries'. Red arcs connect each of these words to the label 'is-a?' on the right, indicating that each word is being evaluated for its relationship to the concept 'country'.

1. The words that are close to the pattern words are most likely correct

$\text{is-a}(\text{China, country})$

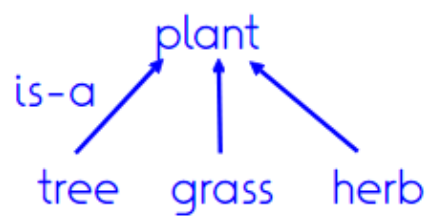
2. If a word has been seen before, then all words between it and the pattern are most likely correct.

Assume we have seen $\text{is-a}(\text{Russia, country})$ before

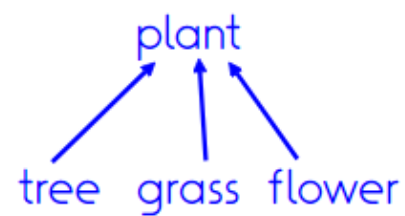
$\Rightarrow \text{is-a}(\text{Brazil, country})$

Distinguishing word senses

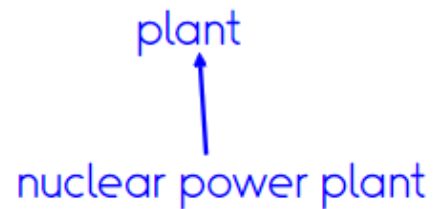
plants such as trees,
grass, and herbs



plants such as trees,
grass, and flowers

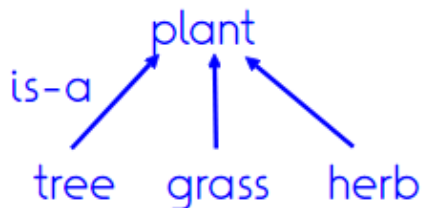


plants such as
nuclear power plants

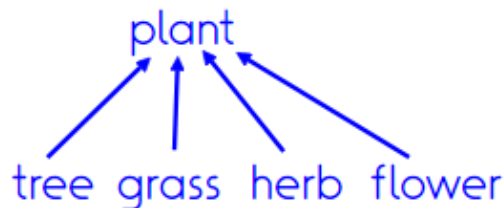
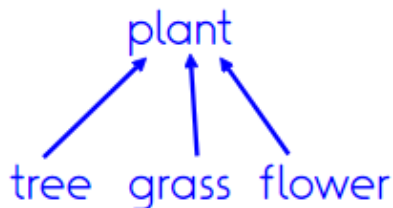


Distinguishing word senses

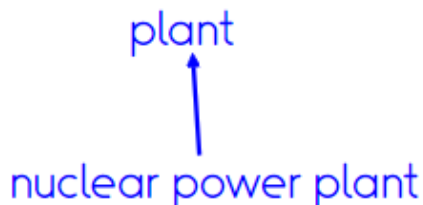
plants such as trees,
grass, and herbs



plants such as trees,
grass, and flowers



plants such as
nuclear power plants



Outline

1. Named-entity recognition and classification
 1. Problem
 2. Features
 3. Rule-based
 4. Distance-based
2. Extractive typing
3. Use cases
 - spaCy
 - WebIsALOD
 - Entyfi

Use case 1: Classification: spaCy

- <https://spacy.io/api/annotation#named-entities>
- Trained on OntoNotes 5 dataset
 - 18 types
 - Few frequent ones (person, location, geopolitical entity, ..)
- <http://localhost:8888/notebooks/IE-course/spacy-NER.ipynb>

```
import spacy
nlp = spacy.load("en_core_web_sm")

text = ("Mary likes icecream. John gives Mary a book. They pay 50 Euro for a ticket.
        They visit the MoMa to watch an exhibition on Aliens. On Tuesday Washington
        negotiates a trade agreement with the Vatican.")
doc = nlp(text)

print([(X.text, X.label_) for X in doc.ents])

from spacy import displacy
displacy.render(doc, style="ent")
```

Use case 2: Extraction: WebIsALOD

- Huge dataset (400m) of noisy isA-edges extracted from general web crawl (CommonCrawl) via 58 Hearst-style patterns

hypernymLabel	hyponymLabel	confidence
"president"	"political leader"	0.816318
"president"	"political figure"	0.777531
"president"	"personality"	0.743645
"president"	"figurehead"	0.739836
"president"	"great man"	0.7374
"president"	"key position"	0.737264
"president"	"important position"	0.734956
"president"	"speaker"	0.731756
"president"	"public office"	0.725221

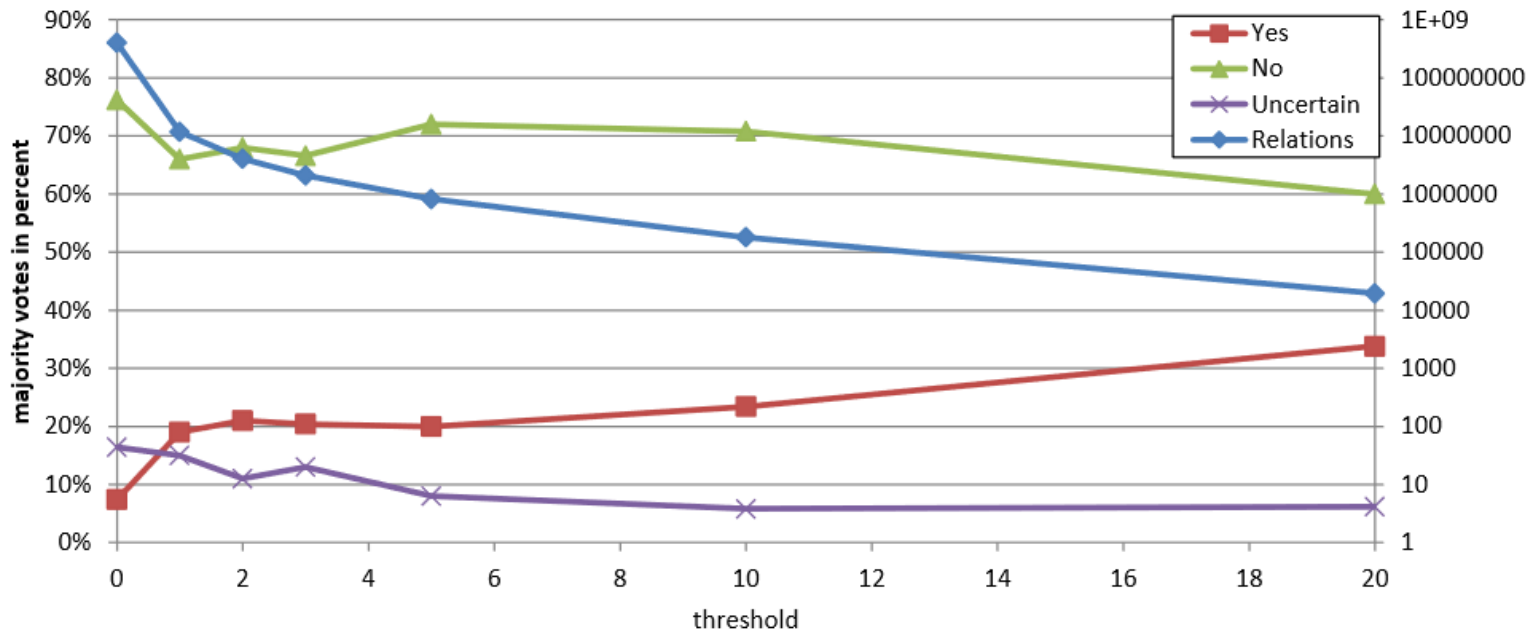
hypernymLabel	hyponymLabel	confidence
"cat"	"family pet"	0.766699
"cat"	"thing in the world"	0.727088
"cat"	"domesticated animal"	0.726393
"cat"	"small creature"	0.72567
"cat"	"common animal"	0.7196
"cat"	"companion animal"	0.695572
"cat"	"creature"	0.693024
"cat"	"brand"	0.679566

pattern_comment
"NPi and other NPc"
"Such NPc as Npi"
"NPi like other NPc "
"NPi, one of the NPc"
"NPi, one of these NPc"
"NPi, one of those NPc"
"Examples of NPc is Npi"
"Examples of NPc are Npi"
"NPi are examples of NPc"
"NPi is example of NPc"
"NPc for example Npi"

[\[http://webisa.webdatacommons.org/\]](http://webisa.webdatacommons.org/)

WebIsALOD - Noise

- Pure application of 58 patterns
 - 75% of results are considered wrong by human annotators
 - Thresholding by source and pattern frequency helps little



WebIsALOD – supervised scoring

- RandomForest trained on 400 instances
- Features:
 - Frequency of matches
 - #distinct patterns
 - #distinct domains
 - Binary feature per pattern
 - Hypernym/hyponym length, modifier existence
 - Min/max/avg token distance in source
 - ROC AUC 0.72-0.83

Data access

<http://webisa.webdatacommons.org/sparql>

```
PREFIX isa: <http://webisa.webdatacommons.org/concept/>
PREFIX isaont: <http://webisa.webdatacommons.org/ontology#>
SELECT ?hyponymLabel ?confidence
WHERE{
  GRAPH ?g {
    isa:_beer_ skos:broader ?hyponym.
  }

  ?hyponym rdfs:label ?hyponymLabel.
  ?g isaont:hasConfidence ?confidence.
}
ORDER BY DESC(?confidence)
```

Use case 3: Joint system: Entyfi

- Existing approaches overfit to Wikipedia, or assume huge corpora
- Fiction and fantasy as archetypes of long-tail universes where Wikipedia has insufficient coverage

“After Melkor’s defeat in the First Age, Sauron became the second Dark Lord and strove to conquer Arda by creating the Rings”

MELKOR: Ainur, Villain

SAURON: Maiar, Villain

RINGS: Jewelry, Magic Things

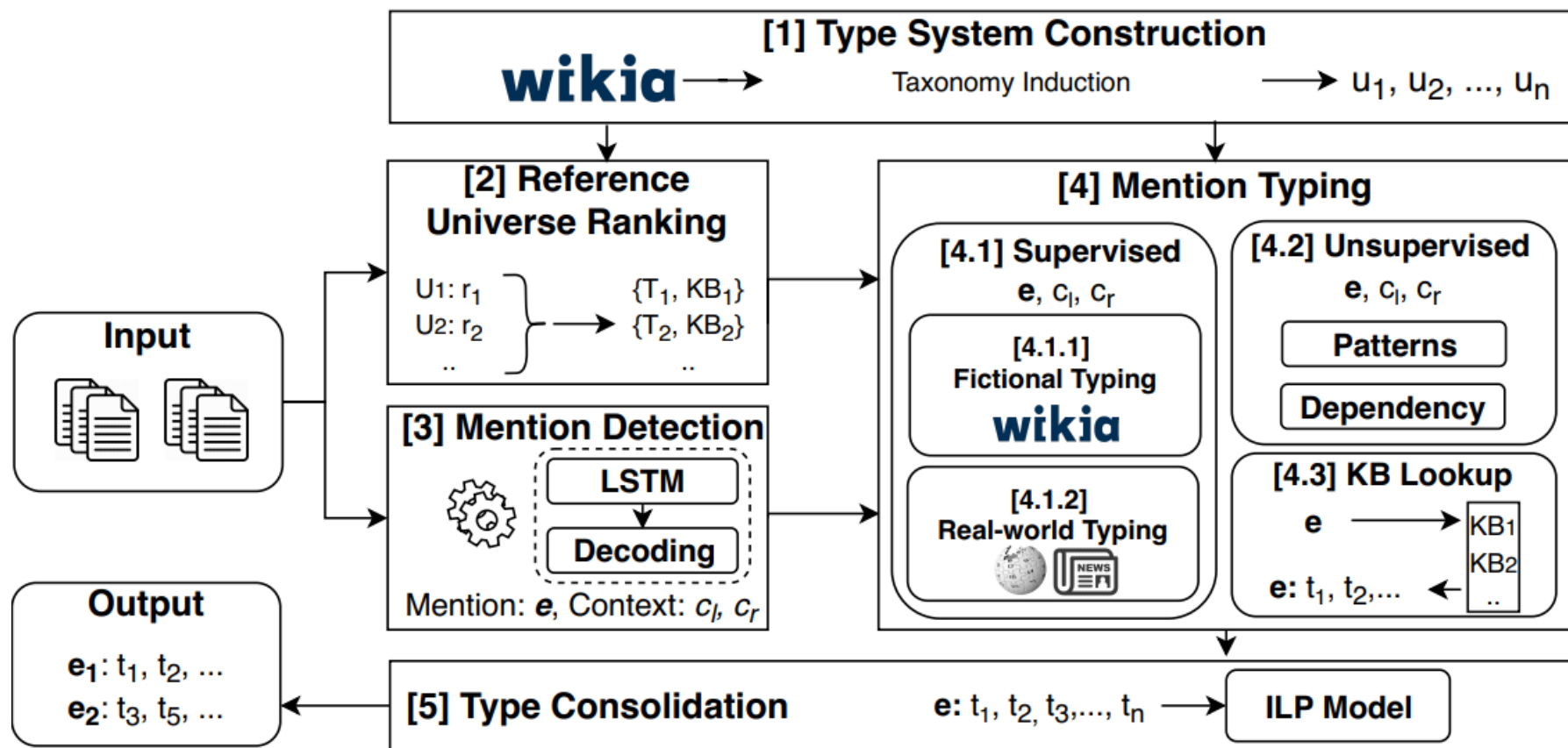
FIRST AGE: Eras, Time

DARK LORD: Ainur, Title

ARDA: Location

[Chu et al., WSDM 2020]

Entify - Overview



Entify – 1. Type system construction

- Scrape categories and subcategory relations from 200 diverse Wikia universes
- Clean extracted data to derive coherent taxonomies
 - Remove meta-categories
 - Remove unsuited relations
 - Forward reference: more details next lecture

Entify – 2. Type system ranking

- Demo:

- First example LoTR+GoT
- The Maya god of rain, one of the most important deities in the Mayan pantheon Panon. Chaac was often described as having four divine aspects or incarnations, connected to the four cardinal directions and colored green, red, white, and black respectively. In some Maya traditions there were also many demigods, also known as Chaacob, who served the great god Chaac and often appeared to humans as dwarves or giants, like Ebxywz.

- <http://halimede:8080/ENTYFI-WEB/>

- (runtime → run 1, 4, 5)

Entify – 4.1 – supervised typing

- Existing Wikipedia models: 112 or 10331 types
 - [Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. Ultra-fine entity typing. In *ACL*, 2018]
 - [Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. Neural architectures for fine-grained entity type classification. In *EACL*, 2017]
- Attentive **neural architectures** on 16-words span around mention
- Trained via **disambiguated links and categories in Wikia** reference universes

Entify – 4.2/4.3 – Unsupervised+lookup

- Unsupervised typing:
 - 36 Hearst patterns
 - Dependency parses
 - King Robert was the ruler of Dragonstone Castle
- Lookups:
 - Fiction frequently creates mashups and reuses previous ideas
 - Freya not only in “Edda”, but also in “Nibelungen”
 - Lookups as practical way to obtain candidates

Entify – 5. Consolidation

- Most typing modules are noisy
- Top-level hard constraints
 - Location and living_thing exclude each other
- Soft correlations
 - Dwarves are frequently axe-wielders, rarely archers
 - Secret agents are usually middle-aged singles
- Consolidation across multiple mentions
 - Gondor decided to join the war.
 - Frodo joined the ranks of Gondor.
- ILP encodes hard constraints and rewards for respecting correlations

Entify - Demo

- <http://halimede:8080/ENTYFI-WEB/>

Enter text: The Lord of the Rings (LOTR) Game of Thrones (GoT) Random Texts

The Maya god of rain, one of the most important deities in the Mayan pantheon Panon. Chaac was often described as having four divine aspects or incarnations, connected to the four cardinal directions and colored green, red, white, and black respectively. In some Maya traditions there were also many demigods, also known as Chaacab, who served the great god Chaac and often appeared to humans as dwarves or giants, like Ebxywz.

Selecting typing module(s). ☒ Select All

☒ Real-world Typing ☒ Supervised Fiction Typing ☒ Unsupervised Typing ☒ KB Lookup Typing ☒ Type Consolidation

Num. real-world types
112 types (quick) ▾

Find Related Universes

Top 5 Reference Universes

Please select reference universe(s).	Similarity Score
☒ Sons of Anarchy	0.5028
☒ Psychology	0.5009
☒ Forgotten Realms	0.5008
☒ Against the Gods	0.5007
☒ Dragon Ball	0.5007

[More Universes](#)

Lab 4

- Input: First sentence from Wikipedia
- Task: Extract/predict the types
- Automated evaluation on unseen evaluation data
 - Precision/recall/F1
 - Macro (per row), micro (per prediction/ground truth type instance)
 - Full match or headword only (British writer/English writer)
 - Concentrate on F1/Macro/headword only
 - How good is good enough? Random baseline
- Hints:
 - Consider data distribution – max bang per buck?
 - Data contains some noise (targets from Wikidata) – don't aim for 100%

References

Sunita Sarawagi: Information Extraction

Diana Maynard: Named Entity Recognition

Sven Hertling and Heiko Paulheim. WebIsALOD: Providing Hypernymy Relations extracted from the Web as Linked Open Data, ISWC 2017

Chu et al., ENTYFI: Entity Typing in Fictional Texts, WSDM 2020

Slides adapted from Fabian Suchanek

Take home

1. NERC first step in making sense of texts
2. Types help in entity organization, fact extraction and search
3. Classification vs. extraction
4. Hearst patterns
5. Use cases: extraction noise and consolidation