INFORMATION RETRIEVAL

ENTITY LINKING, RECOGNITION, AND PART OF SPEECH

lorenzo.bellomo@di.unipi.it - Lorenzo Bellomo

Paolo Ferragina

Part of Speech Tagging

- Maps textual tokens to a set of PoS tags
- Most common resource is the Penn Treebank
- https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

Alphabetical list of part-of-speech tags used in the Penn Treebank Project:

Number	Tag	Description
1.	CC	Coordinating conjunction
2.	CD	Cardinal number
3.	DT	Determiner
4.	EX	Existential *there*
5.	FW	Foreign word
6.	IN	Preposition or subordinating conjunction
7.	JJ	Adjective
8.	JJR	Adjective, comparative
9.	JJS	Adjective, superlative
10.	LS	List item marker
11.	MD	Modal
12.	NN	Noun, singular or mass

Part of Speech Tagging

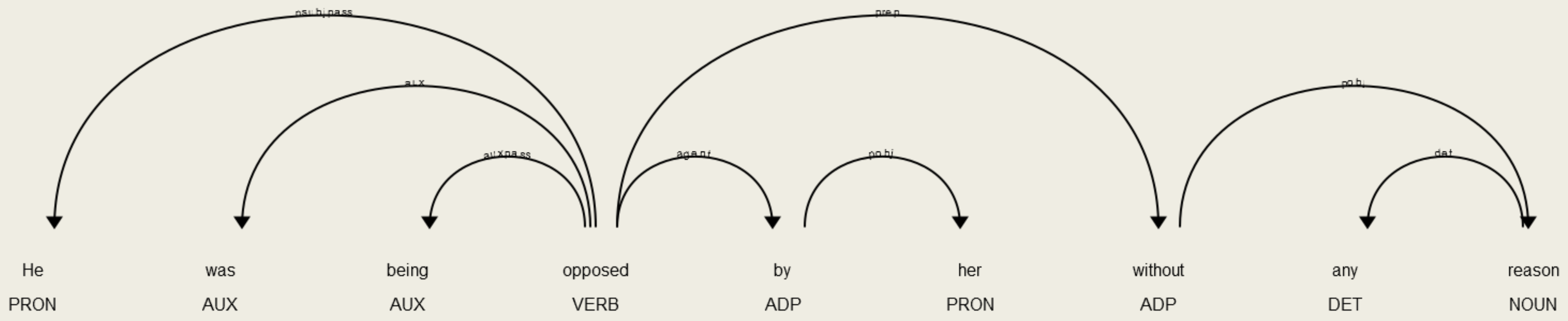
spaCy

```
import spacy

nlp = spacy.load('en_core_web_lg’)

sentence = "He was being opposed by her without any reason.\
            A plan is being prepared by charles for next project"

for token in nlp(sentence):
    print(f'{token.text:{10}} {token.tag_:>{10}}\t{spacy.explain(token.tag_):<{50}}
                    {token.pos_:>{5}}')
```

He	PRP	pronoun, personal	PRON
was	VBD	verb, past tense	AUX
being	VBG	verb, gerund or present participle	AUX
opposed	VBN	verb, past participle	VERB
by	IN	conjunction, subordinating or preposition	ADP
her	PRP	pronoun, personal	PRON
without	IN	conjunction, subordinating or preposition	ADP
any	DT	determiner	DET
reason	NN	noun, singular or mass	NOUN
.	.	punctuation mark, sentence closer	PUNCT
A	DT	determiner	DET
plan	NN	noun, singular or mass	NOUN
is	VBZ	verb, 3rd person singular present	AUX
being	VBG	verb, gerund or present participle	AUX
prepared	VBN	verb, past participle	VERB
by	IN	conjunction, subordinating or preposition	ADP
charles	NNP	noun, proper singular	PROPN
for	IN	conjunction, subordinating or preposition	ADP
next	JJ	adjective (English), other noun-modifier (Chinese)	ADJ
project	NN	noun, singular or mass	NOUN

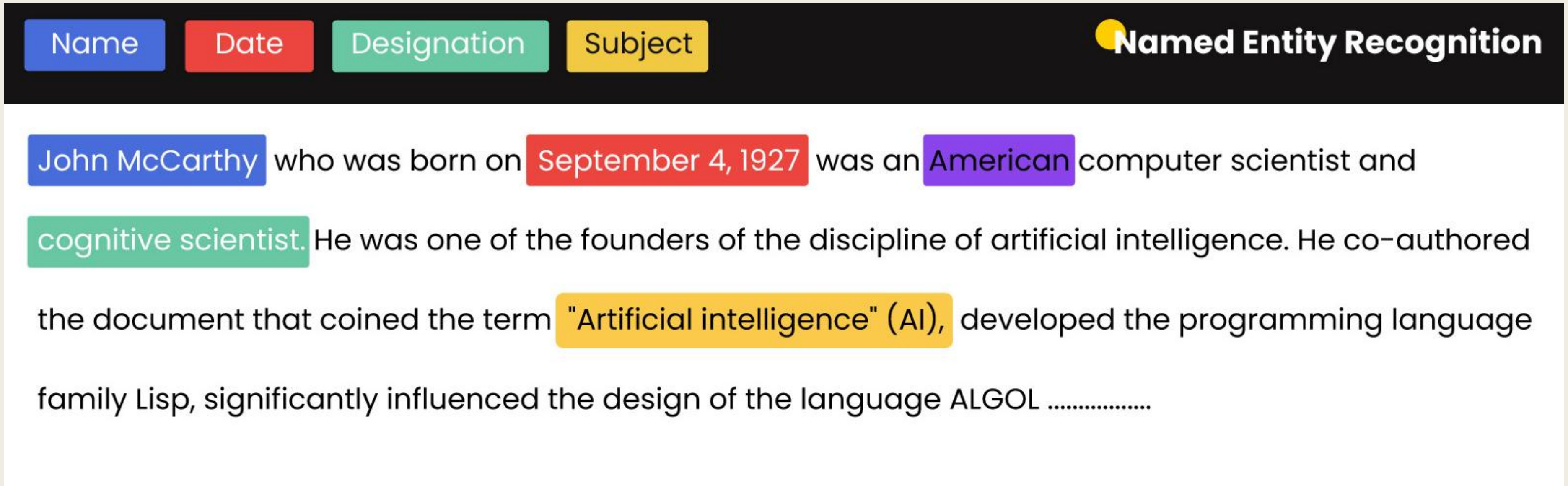
Dependency parsing

```
import spacy

nlp = spacy.load("en_core_web_sm")

doc = nlp("He was being opposed by her without any reason")

spacy.displacy.serve(doc,style="dep")
```



Named Entity Recognition

- A named entity is basically a real-life object
- It has proper identification and can be denoted with a proper name
- It can be a place, person, organization, time, object, geographic entity, or more…



When is it useful?

1. **Information extraction:** usually used to generate reports or to store the retrieved information in specific knowledge bases.
2. **Search engine performance optimization:** NER can be used to tag articles with relevant entities. These can then be stored and used to quickly and efficiently match search queries with relevant articles. This can save computational resources and improve search speed.
3. **Text summarization:** We can use NER to find important named entities in a text or a document and use them to give a summary of the text with contextual information highlighted.

How they work

The NER tools usually fall into one (or more) of these categories:

- **Rule-based:** Rules are manually crafter, and matched on the input text
- **Statistical model:** predict named entities based on likelihoods derived from training data
- **Machine Learning:** They learn from labeled data to predict named entities
- **Deep Learning:** they predict named entities using neural networks
- **Hybrid Methods:** integrate techinques from all the previously mentioned methods

Exercise: perform NER

- First step: annotate text with NER using pre-computed models
- We will use the spaCy NLP suite in Python
  - *pip install spacy*
  - *python -m spacy download en_core_web_sm*
  - *We will use the submodule "displacy" to effectively display the results*

spaCy

Defaults spaCy categories

PERSON,

NORP (nationalities, religious and political groups)

FAC (buildings, airports etc.)

ORG (organizations)

GPE (countries, cities etc.)

LOC (mountain ranges, water bodies etc.)

PRODUCT (products)

EVENT (event names)

WORK_OF_ART (books, song titles)

LAW (legal document titles)

LANGUAGE (named languages)

DATE

TIME

PERCENT

MONEY

QUANTITY

ORDINAL

CARDINAL

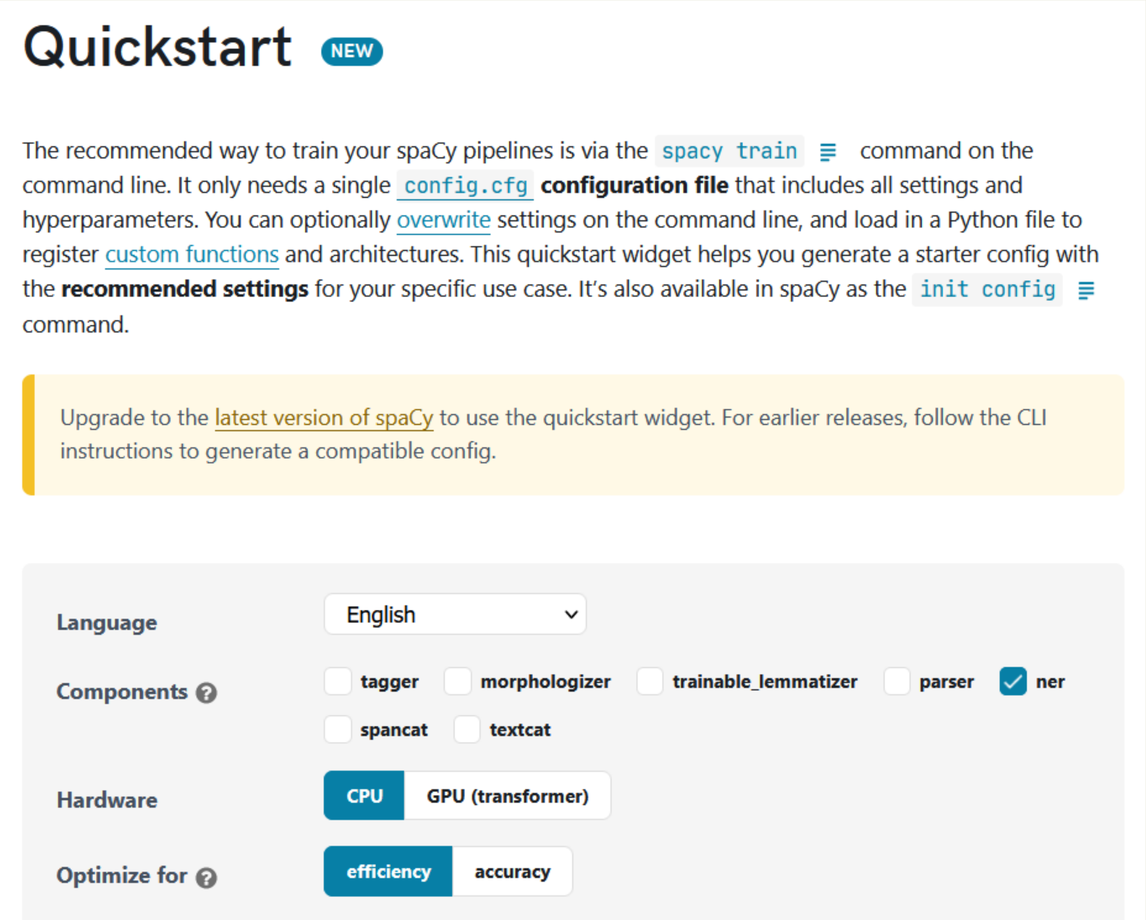
Loading the model

```
import spacy

# Loading the English model
en_model = spacy.load(“en_core_web_sm”)

text = "Apple is looking at buying U.K. startup for $1 billion"
doc = en_model(text)
for entity in doc.ents:
    print(entity.text, entity.label_)
    print(spacy.explain(entity.label_))
```

What if I need different categories?

- What if *people, locations, dates, etc…* are not my focus?
- We can re-train the NER process
  - *This, of course, requires a labeled dataset*
- We will use an example from Biology:
- "The Anatomical Entity Mention (AnEM) corpus"
- https://github.com/juand-r/entity-recognition-datasets/tree/master/data/AnEM
- This dataset consists of abstracts and full-text biomedical papers

```
Entities:

----------------------------------------
Anatomical_system              51
Cell                           776
Cellular_component             199
Developing_anatomical_structure 39
Immaterial_anatomical_entity   60
Multi-tissue_structure         639
Organ                          381
Organism_subdivision           162
Organism_substance             291
Pathological_formation         368
Tissue                         169
```



Quickstart NEW

The recommended way to train your spaCy pipelines is via the `spacy train` command on the command line. It only needs a single `config.cfg` **configuration file** that includes all settings and hyperparameters. You can optionally overwrite settings on the command line, and load in a Python file to register custom functions and architectures. This quickstart widget helps you generate a starter config with the **recommended settings** for your specific use case. It's also available in spaCy as the `init config` command.

Upgrade to the latest version of spaCy to use the quickstart widget. For earlier releases, follow the CLI instructions to generate a compatible config.

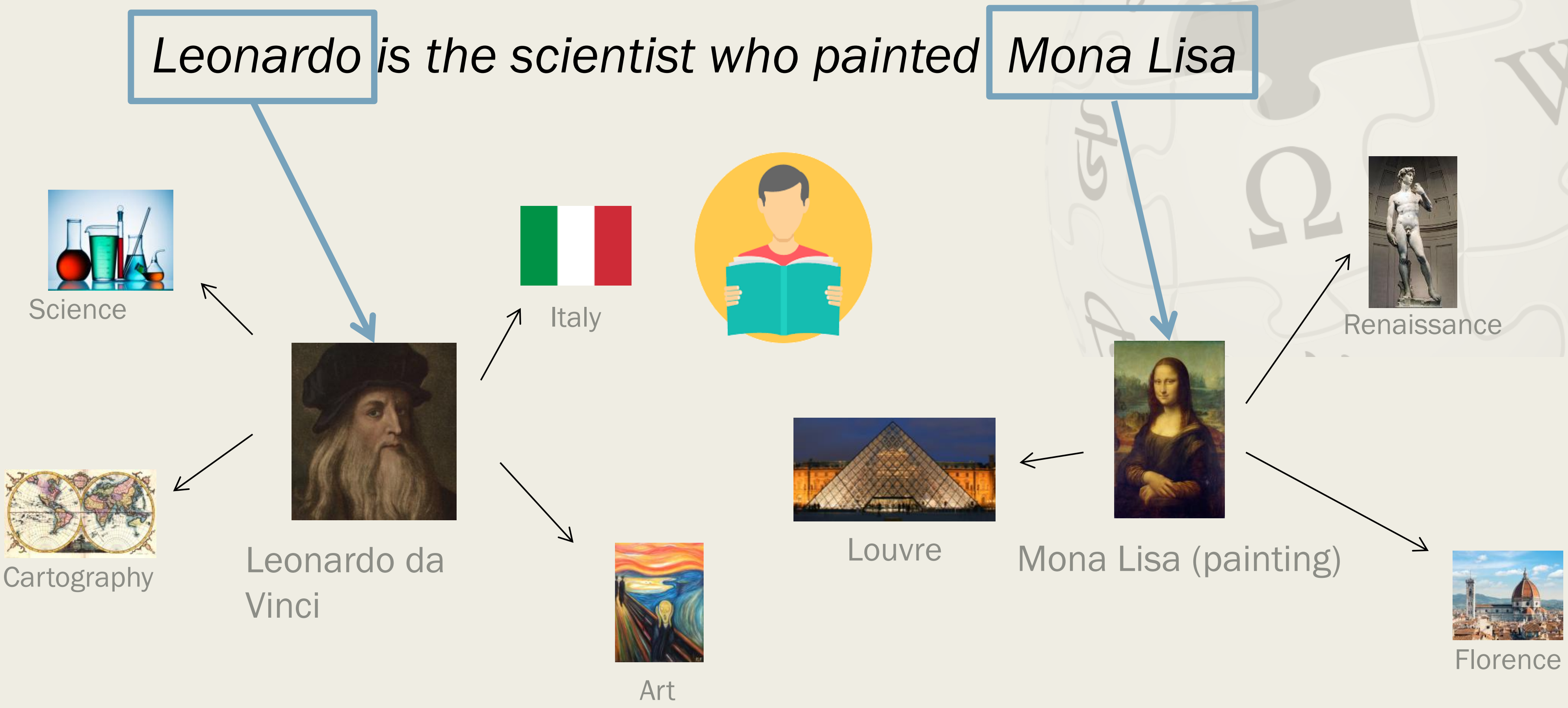
HOW TO TRAIN

https://spacy.io/usage/training #quickstart

Training

```
ℹ Saving to output directory: output
ℹ Using CPU

=========================== Initializing pipeline ===========================
✔ Initialized pipeline

============================= Training pipeline =============================
ℹ Pipeline: ['tok2vec', 'ner']
ℹ Initial learn rate: 0.001
E    #       LOSS TOK2VEC  LOSS NER  ENTS_F  ENTS_P  ENTS_R  SCORE
---  ------  ------------  --------  ------  ------  ------  ------
  0       0          0.00     27.87    0.00    0.00    0.00    0.00
  0     200        308.78   1644.50    8.64   13.06    6.45    0.09
  0     400         85.48    866.24   15.94   33.76   10.43    0.16
  1     600        139.98   1002.82   25.36   33.55   20.38    0.25
  1     800        236.84   1008.65   37.06   50.41   29.30    0.37
  2    1000        237.54    971.08   45.06   53.94   38.69    0.45
  2    1200        730.19    970.37   50.37   59.12   43.87    0.50
  3    1400        477.19    909.08   50.72   58.26   44.90    0.51
  5    1600        453.01    779.01   52.45   68.02   42.68    0.52
  6    1800        469.67    665.57   57.12   66.88   49.84    0.57
  8    2000        461.15    578.49   57.34   69.50   48.81    0.57
✔ Saved pipeline to output directory
output/model-last
```

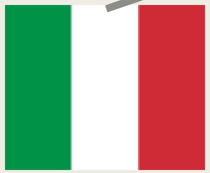
Entity linking



Map ambiguous sequences of words into *real-world* entities as nodes on the graph. Then, *contextualize* them with *related* entities linked in the Knowledge Graph

TagME, disambiguation

"… Diplomatic tension between Italy and Brazil…"



Italy (Nation)

Brazil (Nation)

"…Italy beats Brazil 4-6, tensions between supporters…"

Italy Football Team

Brazil Football Team

From words to Semantics

"…<u>Italy</u> beats <u>Brazil</u> 4-6, tensions between supporters…"

From words to semantics

Brazil

From Wikipedia, the free encyclopedia

Coordinates: 10°S 52°W

"Brazilian Republic" redirects here. For other uses, see Brazil (disambiguation) and Brazilian Republic (disambiguation).

Brazil (Portuguese: *Brasil*; Brazilian Portuguese: [braˈziw]),[nt 4] officially the **Federative Republic of Brazil** (Portuguese: *República Federativa do Brasil*),[9] is the largest country in both South America and Latin America. At 8.5 million square kilometers (3,300,000 sq mi)[10] and with over 214 million people, Brazil is the world's fifth-largest country by area and the seventh most populous. Its capital is Brasília, and its most populous city is São Paulo. The federation is composed of the union of the 26 states and the Federal District. It is the largest country to have Portuguese as an official language and the only one in the Americas;[11][12] it is also

Federative Republic of Brazil

República Federativa do Brasil (Portuguese)

Flag

Coat of arms

Motto: *Ordem e Progresso* (Portuguese)
"Order and Progress"

Anthem: *Hino Nacional Brasileiro* (Portuguese)
"Brazilian National Anthem"

3:22

Brazil (State)

The Entity Linking tools by Acube lab

Welcome to the Tagme Virtual Research Environment. From here, you can access all Entity Linking tools provided by the Acube lab at the University of Pisa.

TAGME

Entity linker, ideal for annotating noisy text.
Available languages: **en**, **de**, **it**

WAT
The Wikipedia Annotator of Text

Entity linker, ideal for annotating well-formed text. More accurate than TagMe, but still experimental.
Available languages: **en**

SMAPH
Linking Queries to Knowledge

Entity linker for web search queries.
Available languages: **en**

SWAT
The Salient Wikipedia Annotator of Text

Entity Salience service: assigns a relevance score to the entities mentioned by a document.
Available languages: **en**

REL

Entity linker, ideal for both short and long text. More accurate than WAT.
Available languages: **en**

Bio TAGME

A comprehensive platform for biological knowledge network analysis.

NetMe
On-the-fly knowledge network construction from biomedical literature

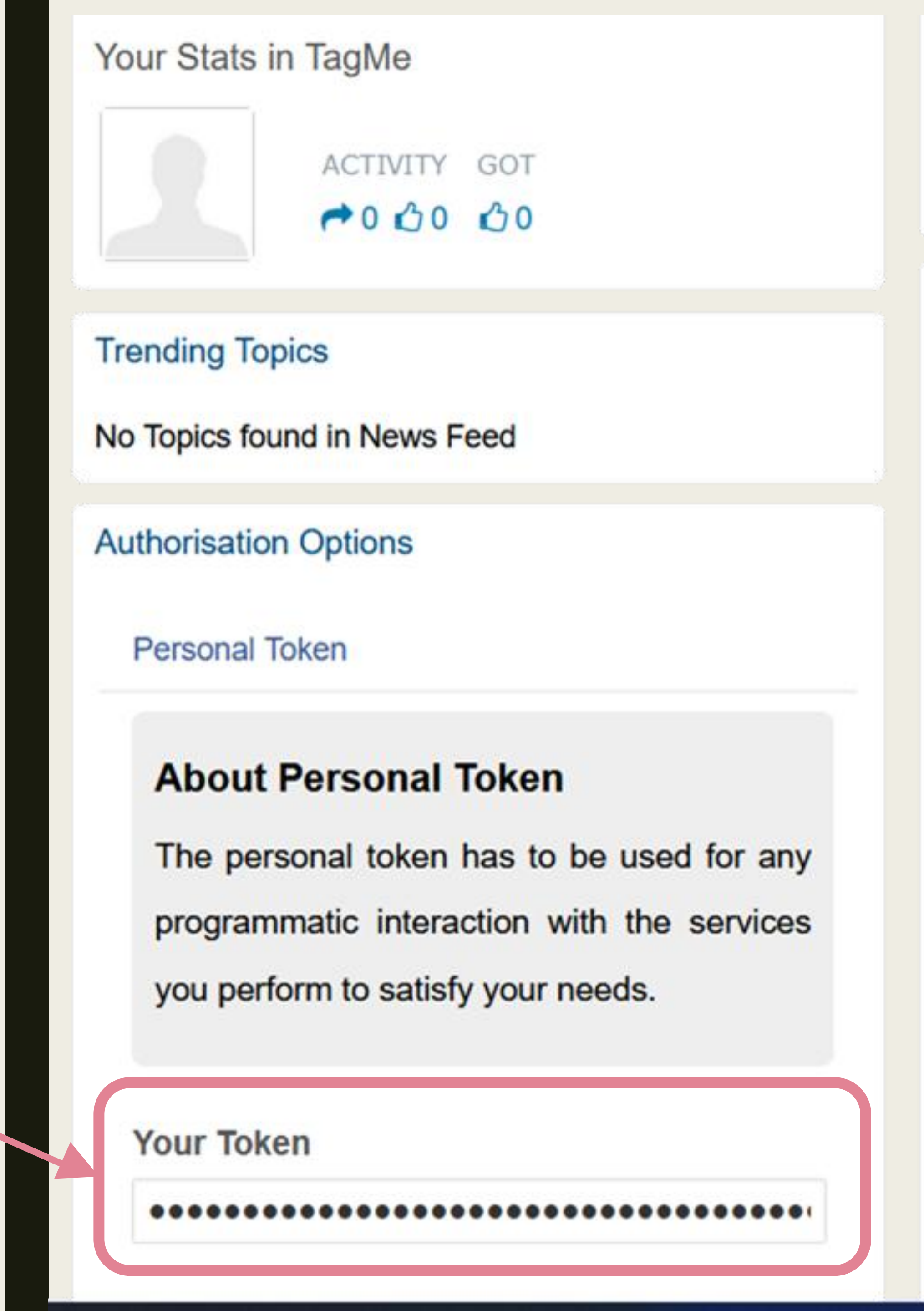
On-the-fly knowledge network construction from biomedical literature.

Exercise: Annotate with TagME

Documentation: https://sobigdata.d4science.org/web/tagme/tagme-help

We need to provide:

- The text to annotate
- The language ("en" – default choice)
- An authentication token (gcube token) – to obtain it go at the url https://sobigdata.d4science.org/group/tagme



Query time

Create annotate.py

```
import requests

TAGME_ENDPOINT = "https://tagme.d4science.org/tagme/tag"
LANG = "en"
KEY = "INSERT YOUR TOKEN HERE"

def query_tagme(text):
    payload = {"text": text, "gcube-token": KEY, "lang": LANG}
    r = requests.post(TAGME_ENDPOINT, payload)
    if r.status_code != 200:
        raise Exception("Error on text: {}\n{}".format(text, r.text))
    return r.json()
```

Response

Italy will not be competing in the 2022 world cup

{'spot': 'Italy’, 'link_probability': 0.44, 'rho': 0.45, 'title': 'Italy national football team’}

{'spot': 'will', 'link_probability': 0.01, 'rho': 0.07, 'title': 'Will (2011 film)’}

{'spot': '2022 world cup', 'link_probability': 0.35, 'rho': 0.34, 'title': '2022 FIFA World Cup’}

Only results with rho > 0.3 are worth keeping

Filter the entities

Add this function to annotate.py:

```
def get_tagme_entities(tagme_response, min_rho=0.3):
    ann = tagme_response["annotations"]
    ann = [a for a in ann if a["rho"] > min_rho]
    return [a["title"] for a in ann if "title" in a]
```

Output will become:
Italy national football team
2022 FIFA World Cup

Fields returned by TagME

- ***spot (string):*** *how the anchor appears in the text*
- ***start (int):*** *the index of the starting character of the anchor*
- ***end (int):*** *the index of the ending character of the anchor*
- ***link_probability (float*** $\in [\mathbf{0}, \mathbf{1}]$***):*** number of times that the **spot** is an anchor in Wikipedia / number of occurrences of the **spot** in Wikipedia
- ***rho (float*** $\in [\mathbf{0}, \mathbf{1}]$***):*** *semantic coherency of the entity with respect to the query*
- ***id (int):*** *the Wikipedia identifier of the page https://en.wikipedia.org/?curid=<>*
- ***title (string):*** *title of the Wikipedia page*

Query long texts with TagME

Modify the function **query_tagme** in **annotate.py**:

```
with open("Leonardo.txt") as long_file:
    text = long_file.read()


def query_tagme(text, long_text=False):
    payload = {"text": text, "gcube-token": KEY, "lang
    if long_text:
        payload["long_text"] = 5
    r = requests.post(ENDPOINT, payload)
    if r.status_code != 200:
        raise Exception("Error on text: {}\n{}".format(text, r.text))
    return r.json()
```

5 is the number of neighboring entities TagME will use to compute the rho-score

Another annotator – SWAT

The Salient Wikipedia Annotator of Text

- Specialized on long input documents.
- It finds the entities and classifies them in salient and non-salient.
- Annotation process is way more involved

Document Enrichment → Feature Generation → Classification

Document Enrichment
- CoreNLP
- TextRank
- WAT

Feature Generation
- Basic
- Syntactic
- Semantic

} Features

Classification

Classify entities in **salient** and **non salient**

The New York Times

POLITICAL ACTION; Decisions on the Horizon

By JEFF ZELENY and PATRICK HEALY

Published: January 9, 2007

Don't look for presidential announcements from Senators Barack Obama and Hillary Rodham Clinton anytime soon, but stay tuned.

At least that is the word from their associates. Mr. Obama, Democrat of Illinois, is not likely to say whether he intends to seek the party's presidential nomination until after President Bush's State of the Union address on Jan. 23. As he walked out of the Capitol on a recent afternoon, Mr. Obama only smiled when asked about his timing. Then, he rushed to change the subject.

Initially, Mr. Obama said he intended to announce his decision after returning from a holiday vacation in Hawaii, where he was visiting his grandmother and other relatives. Now, several people close to the senator say, he needs a little more time to make up his mind.

Still, Mr. Obama has been busy telephoning crucial Democrats in Iowa, New Hampshire and other states. There is, of course, only one reason for him to be making such inquiries.

Last week on Capitol Hill, Mr. Obama bumped into Ethel Kennedy, who has been a big admirer. When asked about him, she said, "He can't run soon enough."

Mrs. Clinton, meanwhile, plans to announce her decision in the next several weeks, her advisers say. According to several Democrats who have spoken to her, as well as advisers, Mrs. Clinton has given every indication that she is running, short of saying so, and no signals that she is not.

She is making phone calls to Democratic officials, labor leaders and supporters in early nominating states. And she continues to talk to possible consultants and donors, yet she has not made any travel plans to kick off a campaign. JEFF ZELENY and PATRICK HEALY

Entity	Salience
Barack_Obama	1
Hillary_Clinton	1
Hawaii	0
George_W._Bush	0
...	...

Query SWAT

Modify annotate.py

```
import json
SWAT_ENDPOINT = "https://swat.d4science.org/salience"

def query_swat(title, content):
    document = json.dumps({"title": title, "content": content})
    r = requests.post(SWAT_ENDPOINT, data=document, params={"gcube-token": KEY})
    if r.status_code != 200:
        raise Exception("Error on text: {}\n{}".format(text, r.text))
    return r.json()["annotations"]
```

Fields returned by SWAT

- **salience_class (int)**: 1 if the entity is deemed salient, 0 otherwise
- **salience_score (float $\in [\mathbf{0}, \mathbf{1}]$):** the saliency of the enitity in the text (similar to the *rho-score* in tagme)
- **spans (list):** list of times where this entity appears, they are described as:
  - ***start (int):*** *the index of the starting character of the anchor*
  - ***end (int):*** *the index of the ending character of the anchor*
- **wiki_id (int):** the Wikipedia identifier of the page
- **wiki_title (string):** title of the Wikipedia page

What to do with annotations

- Usually the best way to exploit annotations is to build **graphs**
- We can **link entities** together using a weighted graph
  - *Nodes are the entities*
  - *Edges are weighted and their weight is their relatedness*
- If the edges are few can ask TagME for the relatedness according to a well known metric (*Milne&Witten*)
- Otherwise we can exploit **embedding vectors**

Query entity relatedness

Create a new file relatedness.py

```
ENDPOINT_RELATEDNESS = "https://tagme.d4science.org/tagme/rel"

def query_relatedness(e1, e2):
    tt = e1.replace(" ", "_") + " " + e2.replace(" ", "_")
    payload = {"tt": tt, "gcube-token": KEY, "lang": LANG}

    r = requests.post(ENDPOINT_RELATEDNESS, payload)
    if r.status_code != 200:
        raise Exception("Error on relatedness computation: {}\n{}".format(tt, r.text))
    return r.json()
```

References

- *TAGME: On-the-fly Annotation of Short Text Fragments (by Wikipedia Entities)*
  https://www.researchgate.net/publication/220269845_TAGME_On-the-fly_annotation_of_short_text_fragments_by_Wikipedia_entities
- *SWAT: A system for detecting salient Wikipedia entities in texts*
  https://arxiv.org/abs/1804.03580
- *REL: An Entity Linker Standing on the Shoulders of Giants*
  https://arxiv.org/abs/2006.01969
- *Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia*
  https://arxiv.org/abs/1812.06280