



Python per la Manipolazione dei Testi

Parte 3

Cezar Sas

University of Groningen

c.a.sas@rug.nl



Pipeline Analisi Testo



—

spaCy



Analisi del Testo

Google spinge su Stadia: sarà installata di default sui Chromebook

- Di cosa si parla?
- Chi fa cosa a chi?
- Che aziende o persone vengono menzionati?
- ...



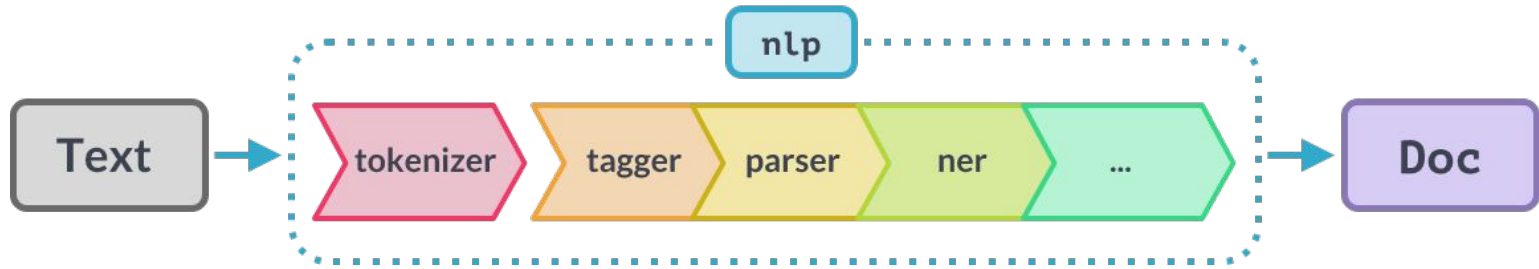
Spacy

- Libreria NLP open-source
- Pronta per la produzione
- Offre funzionalità di/per il deep learning
- Multilingua

```
pip install spacy
```


Pipeline

```
python -m spacy download it_core_news_sm
```





Pipeline



```
import spacy

nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying U.K. startup for $1 billion")
```

Funzionalità

(carrellata di termini di grammatica)



Tokenizzazione

Separare una sequenza in parole, punteggiatura, ecc.

Google spinge su Stadia: sarà installata di default sui Chromebook

Google | spinge | su | Stadia | : | sarà | installata | di | default | sui | Chromebook



Analisi Grammaticale

Annotare con la categoria lessicale le parole di un Documento utilizzando un modello statistico

Google | spinge | su | Stadia | : | sarà | installata | di | default | sui | Chromebook

PROPN | VERB | ADP | PROPN | PUNCT | AUX | VERB | ADP | NOUN | ADP | PROPN



Analisi Grammaticale

```
import spacy

nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying U.K. startup for $1 billion")

for token in doc:
    print(token.text, token.lemma_, token.pos_, token.tag_, token.dep_,
          token.shape_, token.is_alpha, token.is_stop)
```



Analisi Grammaticale

TEXT	LEMMA	POS	TAG	DEP	SHAPE	ALPHA	STOP
Apple	apple	PROPN	NNP	nsubj	Xxxxx	True	False
is	be	AUX	VBZ	aux	xx	True	True
looking	look	VERB	VBG	ROOT	xxxx	True	False
at	at	ADP	IN	prep	xx	True	True
buying	buy	VERB	VBG	pcomp	xxxx	True	False
U.K.	u.k.	PROPN	NNP	compound	X.X.	False	False
startup	startup	NOUN	NN	dobj	xxxx	True	False
for	for	ADP	IN	prep	xxx	True	True
\$	\$	SYM	\$	quantmod	\$	False	False
1	1	NUM	CD	compound	d	False	False
billion	billion	NUM	CD	pobj	xxxx	True	False



Morfologia

Spacy permette di fare l'analisi per capire come stato `Fless` o un `Lemma` per ottenere la parola analizzata.

Google | spinge | su | Stadia | : | sarà | installata | di | default | sui | Chromebook

spinge: Mood=Ind | Number=Sing | Person=3 | Tense=Pres | VerbForm=Fin

sarà: Mood=Ind | Number=Sing | Person=3 | Tense=Fut | VerbForm=Fin

sui: Definite=Def | Gender=Masc | Number=Plur | PronType=Art



Morfologia

```
import spacy

nlp = spacy.load("en_core_web_sm")
print("Pipeline:", nlp.pipe_names)
doc = nlp("I was reading the paper.")
token = doc[0] # 'I'
print(token.morph) # 'Case=Nom|Number=Sing|Person=1|PronType=Prs'
print(token.morph.get("PronType")) # ['Prs']
```



Estrazione Lemma

Data una parola, estrarre la sua forma base (Lemma). Ad esempio dato **sarà** la forma base e' essere.

Google | spinge | su | Stadia | : | sarà | installata | di | default | sui | Chromebook

Google | spingere | su | Stadia | : | essere | installare | di | default | sul | Chromebook



Estrazione Lemma

```
import spacy

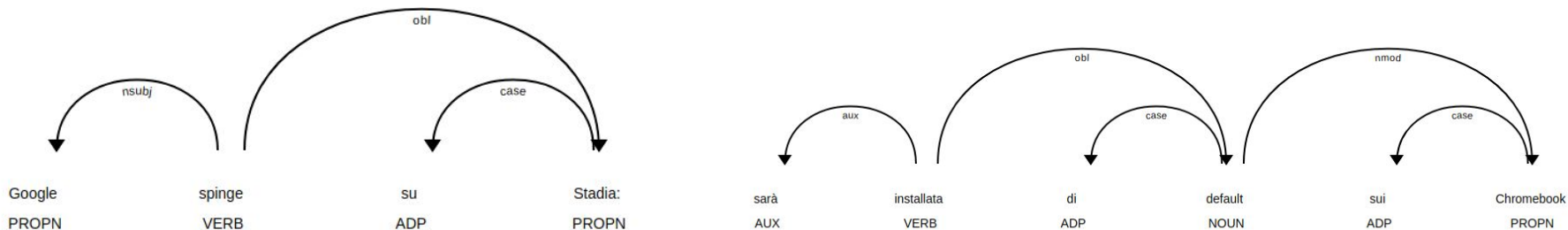
# English pipelines include a rule-based lemmatizer
nlp = spacy.load("en_core_web_sm")
lemmatizer = nlp.get_pipe("lemmatizer")
print(lemmatizer.mode) # 'rule'

doc = nlp("I was reading the paper.")
print([token.lemma_ for token in doc])
# ['I', 'be', 'read', 'the', 'paper', '.']
```


Grammatica Valenziale

Secondo la grammatica valenziale al centro della frase c'è il verbo e ad esso sono attaccati gli argomenti, cioè gli elementi strettamente necessari per completare il significato del verbo stesso.

Google | spinge | su | Stadia | : | sarà | installata | di | default | sui | Chromebook





Grammatica Valenziale

```
import spacy
from spacy import displacy

nlp = spacy.load("it_core_news_sm")
doc = nlp("Google spinge su Stadia: sarà installata di default sui Chromebook")
# Since this is an interactive Jupyter environment, we can use displacy.render here
displacy.render(doc, style='dep')
```



Riconoscimento delle Entità

Estrarre e categorizzare le `Entità` menzionate all'interno del testo.

Google `[MISC]` | spinge | su | Stadia `[LOC]` | : | sarà | installata |
di | default | sui | Chromebook `[ORG]`

Riconoscimento delle Entità

```
import spacy

nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying U.K. startup for $1 billion")

for ent in doc.ents:
    print(ent.text, ent.start_char, ent.end_char, ent.label_)
```

Apple **ORG** is looking at buying U.K. **GPE** startup for \$1 billion **MONEY**



Linking delle Entita

Collegare le `Entita` identificate ad una base di conoscenza (va addestrato).

Google [Q95] | spinge | su | Stadia [Q60309635] | : | sarà | installata |
di | default | sui | Chromebook [Q861508]



Linking delle Entita

```
import spacy

nlp = spacy.load("my_custom_el_pipeline")
doc = nlp("Ada Lovelace was born in London")

# Document level
ents = [(e.text, e.label_, e.kb_id_) for e in doc.ents]
print(ents) # [('Ada Lovelace', 'PERSON', 'Q7259'), ('London', 'GPE', 'Q84')]

# Token level
ent_ada_0 = [doc[0].text, doc[0].ent_type_, doc[0].ent_kb_id_]
ent_ada_1 = [doc[1].text, doc[1].ent_type_, doc[1].ent_kb_id_]
ent_london_5 = [doc[5].text, doc[5].ent_type_, doc[5].ent_kb_id_]
print(ent_ada_0) # ['Ada', 'PERSON', 'Q7259']
print(ent_ada_1) # ['Lovelace', 'PERSON', 'Q7259']
print(ent_london_5) # ['London', 'GPE', 'Q84']
```

Linking delle Entita

```
import spacy

nlp = spacy.load("my_custom_el_pipeline")
doc = nlp("Ada Lovelace was born in London")

# Document level
ents = [(e.text, e.label_, e.kb_id_) for e in doc.ents]
print(ents) # [('Ada Lovelace', 'PERSON', 'Q7259'), ('London', 'GPE', 'Q84')]

# Token level
ent_ada_0 = [doc[0].text, doc[0].ent_type_, doc[0].ent_kb_id_]
ent_ada_1 = [doc[1].text, doc[1].ent_type_, doc[1].ent_kb_id_]
ent_london_5 = [doc[5].text, doc[5].ent_type_, doc[5].ent_kb_id_]
print(ent_ada_0) # ['Ada', 'PERSON', 'Q7259']
print(ent_ada_1) # ['Lovelace', 'PERSON', 'Q7259']
print(ent_london_5) # ['London', 'GPE', 'Q84']
```

TRAIN
IT
YOURSELF

Matching a regole

(o ricerca con regex in spaCy)



Match di token

LIVE



Match di dipendenze

LIVE