



# Confronto tra diversi tool di Named Entity Recognition su testo in linguaggio naturale

Elaborato di Security and Knowledge Management (6 CFU)

## Indice

<b>1</b>	<b>Introduzione</b>	<b>2</b>
<b>2</b>	<b>Named Entity Recognition (NER)</b>	<b>2</b>
<b>3</b>	<b>Estrattori di entità</b>	<b>4</b>
3.1	Dandelion . . . . .	4
3.2	Natural Language framework . . . . .	5
3.3	Polyglot . . . . .	6
3.4	Spacy . . . . .	7
3.5	Natural Language Toolkit (NLTK) . . . . .	7
3.6	Stanford CoreNLP . . . . .	8
3.7	Stanza . . . . .	9
3.8	Tint . . . . .	9
3.9	Google Cloud . . . . .	10
<b>4</b>	<b>Dataset scelti</b>	<b>11</b>
4.1	Dataset in lingua inglese . . . . .	11
4.2	Dataset in lingua italiana . . . . .	13
<b>5</b>	<b>Esperimenti realizzati</b>	<b>15</b>
<b>6</b>	<b>Risultati ottenuti</b>	<b>18</b>
6.1	Valutazione della performance . . . . .	18
6.2	Tempi di computazione . . . . .	20
6.3	Conclusioni e sviluppi futuri . . . . .	21

# 1 Introduzione

Lo scopo di questo elaborato è quello di valutare e confrontare diversi tool per l'estrazione di entità nominate in testi sia in inglese che in italiano.

Nel capitolo 2 viene introdotto il processo di **NER** (*Named Entity Recognition*) e viene fornita una panoramica sui principali approcci al problema, e il capitolo 3 illustra brevemente ciascuno dei tool utilizzati.

Nei capitoli 4 e 5 vengono descritti rispettivamente i due dataset utilizzati per gli esperimenti (inglese e italiano), i passaggi di preprocessing dei dati e gli esperimenti realizzati. Infine, il capitolo 6 mostra i risultati ottenuti.

## 2 Named Entity Recognition (NER)

L'elaborazione del linguaggio naturale (o **NLP**, **Natural Language Processing**), è un ambito dell'intelligenza artificiale che si occupa dei problemi legati all'analisi e all'elaborazione dei linguaggi naturali. [1]

Al contrario dei linguaggi formali, utilizzati nell'interazione tra uomo e macchina, i linguaggi naturali non sono definiti da regole precise che ne permettono un'interpretazione univoca, ma presentano delle caratteristiche di **ambiguità** (eccezioni, sinonimi, figure retoriche, forme idiomatiche dipendenti fortemente dal contesto ecc...) che rendono il processo di elaborazione difficile e complesso.

La ricerca in ambito NLP è rivolta principalmente allo studio di meccanismi di risoluzione di queste ambiguità, e di come possano essere riprodotti tramite algoritmi eseguibili dalle macchine.

Alcuni dei campi di cui si occupa NLP sono:

- *Automatic Summarization*: la produzione automatica di un riassunto dei contenuti principali a partire da un testo;
- *Information Extraction*: l'estrazione di informazioni semantiche da dati non strutturati, come ad esempio documenti di testo;
- *Optical Character Recognition*: la ricostruzione di un testo tramite il rilevamento dei caratteri che lo compongono, a partire da un'immagine;
- *Sentiment Analysis*: la costruzione di sistemi per l'identificazione ed estrazione di opinioni a partire da un testo;
- *Speech Recognition*: sistemi di riconoscimento vocale per la trascrizione automatica di testo a partire da un file audio.

Fa parte dell'*Information Extraction* anche l'*estrazione di entità con nome* (o **Named Entity Recognition**, **NER**), un processo che cerca di trovare e classificare ogni singolo elemento presente in un testo all'interno di categorie predefinite come, ad esempio, persone, organizzazioni, luoghi o eventi. Questo processo è suddiviso come due problemi distinti: l'individuazione delle entità in una frase e la loro classificazione in base alle categorie alle quali appartengono. [2]

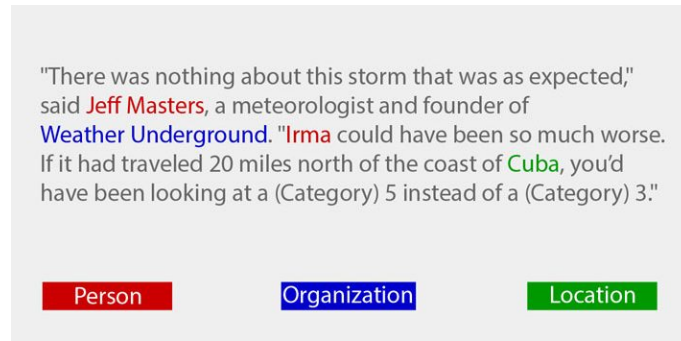


Figura 1: Esempio di Named Entity Recognition in un testo in lingua inglese. (Fonte: *opendatascience.com*)

I possibili approcci a entrambi i problemi possono essere di tre tipi:

- Approccio basato su **lookup list**, quindi sul riconoscimento di parti del testo all'interno di liste predefinite suddivise per categorie. È un sistema semplice, ma che richiede liste estremamente complesse ed estese di possibili entità, difficili da mantenere;
- Approccio basato su una **lista di regole** (*rule-based*), in cui il testo viene analizzato secondo un insieme di regole grammaticali e sintattiche che determinano la presenza di entità all'interno di un testo. Come già illustrato in precedenza, la complessità e l'irregolarità del linguaggio umano rende molto difficile la stesura di regole che siano efficaci per qualsiasi contesto;
- Approccio **statistico**, attraverso l'uso di tecniche di **machine learning**. Questo approccio consiste nell'identificare *pattern* specifici nei dati forniti in input per determinare in modo probabilistico dove potrebbe essere presente un'entità.

L'introduzione di tecniche di Machine Learning nel NLP, avvenuta intorno alla fine degli anni '80, e in particolare la possibilità di applicare modelli statistici all'analisi del linguaggio ha rappresentato una vera e propria rivoluzione nel settore. Un tipico approccio, basato su ML, consiste nel vedere NER come un problema di *sequence tagging* per la categorizzazione, attraverso l'assegnamento di etichette degli elementi che compongono una data sequenza. Alcuni esempi di modelli su cui si basano gli algoritmi utilizzati per questo tipo di task sono *Hidden Markov Models*, *Maximum Entropy Markov Models*, *Conditional Random Fields* e *Support Vector Machines*. [3]

L'obiettivo di questo elaborato, quindi, è quello di provare e confrontare diversi tool per l'estrazione di entità, classificandoli in base alla performance e ai risultati ottenuti sullo stesso set di dati. Il capitolo 3 fornisce una panoramica su ciascuno dei tool utilizzati.

## 3 Estrattori di entità

Per questo elaborato sono stati individuati nove tool diversi per il riconoscimento di entità, elencati di seguito.

### 3.1 Dandelion

Dandelion (*dandelion.eu*) è una piattaforma web che offre servizi di analisi semantica su testi in linguaggio naturale. [4]

Al momento lavora su testi in inglese, francese, tedesco, spagnolo, portoghese e italiano. Sono messe a disposizione diverse funzioni:

- *Estrazione di entità*: trova menzioni di luoghi, persone, marchi ed eventi in documenti e social media. Ottieni facilmente ulteriori dati sulle entità;
- *Classificazione di testo e contenuto*: Classificare il testo multilingue in tassonomie standard predefinite o creare il proprio schema di classificazione personalizzato in pochi minuti;
- *Analisi del sentimento*: Identificare se l'opinione espressa nei brevi testi (come le recensioni dei prodotti) è positiva, negativa o neutra;
- *Similarità semantica*: confronta due testi e calcola la loro somiglianza sintattica e semantica, utile per capire ad esempio se riguardano lo stesso argomento.

In questo elaborato è stata utilizzata esclusivamente la funzione di estrazione di entità. Il servizio è gratuito per un uso limitato (fino a mille richieste al giorno), ed offre invece dei piani a pagamento per le aziende (a partire da 49\$ al mese).

Le API messe a disposizione richiedono un token di autenticazione, che viene indicato nella pagina del profilo dopo essersi registrati alla piattaforma web.

L'endpoint per le API è il seguente:

*<https://api.dandelion.eu/datatxt/nex/v1/>*

E supporta i seguenti parametri:

- *text*: testo da analizzare (o, in alternativa, *html* per passare direttamente codice html o *url* per estrarre testo da un indirizzo web);
- *token*: il token di autenticazione;
- *lang*: lingua del testo da analizzare (de | en | es | fr | it | pt | ru | auto);
- *top\_entities*: specifica il numero delle entità più importanti che devono essere incluse nella risposta. Se questo valore è maggiore di zero, verrà applicato un algoritmo di classificazione per ordinare tutte le entità in base alla loro importanza rispetto al testo di input e solo le più importanti verranno incluse nella risposta;
- *min\_confidence*: specifica la soglia per il valore di confidenza: le entità con un valore di confidenza inferiore a questa soglia verranno eliminate. La fiducia è una stima numerica della qualità dell'annotazione, che varia tra 0 e 1. Una soglia più alta significa che si otterranno meno annotazioni ma più precise. Per questo elaborato è stato scelto di utilizzare un valore fissato a 0.75;

- *social.hashtag*: serve per abilitare il supporto agli hashtag (#), per analizzare correttamente tweet e post di *Facebook*;
- *social.mention*: come per gli hashtag, ma abilita il supporto alle menzioni (@);
- *include*: lista separata da virgola che indica quali ulteriori informazioni restituire sulle entità annotate: ad esempio "types" aggiunge informazioni sul tipo di entità da *DBpedia*, "abstract" aggiunge il testo dell'estratto di *Wikipedia*, "image" aggiunge un link a un'immagine dell'entità taggata, "lod" aggiunge collegamenti a entità equivalenti (sameAs) nei repository supportati di *Linked Open Data*.

```
import requests

token = "XXX" # token di autenticazione
text = "Parigi e' la capitale della Francia." # testo da analizzare
base_url = 'https://api.dandelion.eu/datatxt/nex/v1/?lang=it \
&min_confidence=0.75&token={}&text={}'.format(token, text)
json = requests.get(base_url).json()
list(set([entity['spot'] for entity in json['annotations']]))

# Output: ['Parigi', 'Francia']
```

1: Esempio di codice Python per l'estrazione di entità tramite le API di *Dandelion*.

### 3.2 Natural Language framework

Natural Language è un framework nativo di Apple progettato che fornisce delle API per attività di elaborazione di linguaggio naturale su tutti i dispositivi proprietari. [5]

Il framework *Natural Language* fornisce una varietà di funzionalità di elaborazione del linguaggio con supporto per tante lingue, tra cui inglese e italiano. Alcune di queste funzionalità sono:

- *Identificazione della lingua*, rilevamento automatico della lingua da un testo;
- *Tokenizzazione*, suddivisione di un testo in unità linguistiche o token;
- *Etichettatura delle parti del discorso*, marcatura di singole parole con la loro parte del discorso;
- *Lemmatizzazione*, per ottenere la forma canonica di una parola in base alla sua analisi morfologica;
- *Riconoscimento delle entità nominate*, identificando i token come nomi di persone, luoghi o organizzazioni;
- È inoltre possibile utilizzare questo framework per addestrare e distribuire modelli di linguaggio naturale personalizzati.

Il framework richiede l'installazione del linguaggio di programmazione *Swift*, compatibile al momento solo con sistemi *macOS* e *Linux*.

```

import NaturalLanguage

// testo da analizzare
var text: String = "Parigi e' la capitale della Francia."

var entities: [String] = []
let tagger = NLTagger(tagSchemes: [.nameType])
tagger.string = text
tagger.enumerateTags(in: text.startIndex..

```

2: Esempio di codice Swift per l'estrazione di entità tramite il framework *Natural Language*.

### 3.3 Polyglot

Polyglot è un altro tool open-source scritto in linguaggio *Python* per l'elaborazione del linguaggio naturale con supporto ad un grande numero di lingue diverse, tra cui inglese e italiano. [6]

Supporta diverse funzionalità, tra cui:

- Tokenizzazione (165 lingue);
- Rilevamento della lingua (196 lingue);
- Riconoscimento delle entità nominate (40 lingue);
- Etichettatura delle parti del discorso (16 lingue);
- Analisi del sentimento (136 lingue);
- Analisi morfologica (135 lingue);
- Traslitterazione (69 lingue).

Per il riconoscimento di entità, Polyglot riconosce 3 categorie di entità:

- **Sedi** (Tag: *I-LOC*): città, paesi, regioni, continenti, quartieri, divisioni amministrative...;
- **Organizzazioni** (Tag: *I-ORG*): squadre sportive, giornali, banche, università, scuole, organizzazioni non profit, aziende...;
- **Persone** (Tag: *I-PER*): politici, scienziati, artisti, atleti...

```
from polyglot.text import Text

text = "Parigi e' la capitale della Francia." # testo da analizzare
blob = Text(text)
list(set([' '.join(entity) for entity in blob.entities]))

# Output: ['Parigi', 'Francia']
```

3: Esempio di codice Python per l'estrazione di entità con *Polyglot*.

### 3.4 Spacy

spaCy è una libreria open source tra le più usate per l'elaborazione del linguaggio naturale, scritta in *Python* e rilasciata sotto licenza *MIT*. [7]

Attualmente *Spacy* implementa modelli statistici di reti neurali in inglese, tedesco, spagnolo, portoghese, francese, italiano, olandese e greco; inoltre offre funzionalità di *NER* e di tokenizzazione per diverse altre lingue. Supporta molte funzionalità, tra cui:

- Tokenizzazione;
- Riconoscimento delle entità nominate;
- Supporto per oltre 59 lingue;
- 46 modelli statistici per 16 lingue;
- Etichettatura delle parti del discorso;
- Visualizzatori integrati per sintassi e *NER*;
- Facile integrazione con deep learning (supporto all'integrazione con *TensorFlow*, *PyTorch*, *scikit-learn* e *Gensim*).

I suoi punti di forza sono la velocità, efficienza e la possibilità di costruire facilmente modelli statistici linguisticamente sofisticati per una varietà di problemi di *NLP*.

```
import spacy

text = "Parigi e' la capitale della Francia." # testo da analizzare
nlp = spacy.load('it_core_news_sm') # carica modello italiano
doc = nlp(text)
list(set([e.text for e in doc.ents]))

# Output: ['Parigi', 'Francia']
```

4: Esempio di codice Python per l'estrazione di entità con *Spacy*.

### 3.5 Natural Language Toolkit (NLTK)

NLTK (Natural Language Toolkit), è una suite di moduli *Python* open source, dataset e tutorial che supportano la ricerca e lo sviluppo nell'elaborazione del linguaggio naturale. [8]

*NLTK* fornisce un'implementazione degli algoritmi più comuni in *NLP* come la tokenizzazione, etichettatura di parti del discorso, stemming, analisi del sentimento, segmentazione e riconoscimento di entità nominate. È uno strumento molto potente che però non supporta la lingua italiana (almeno per il riconoscimento di entità).

*NLTK* tra le altre cose punta anche supportare la ricerca e l'insegnamento dell'*NLP* e di altri campi correlati, come la linguistica, le scienze cognitive, l'intelligenza artificiale, l'*information retrieval*, e il *machine learning*; è stato usato con successo come ausilio all'insegnamento, come strumento per lo studio individuale e come piattaforma per prototipare e sviluppare strumenti di ricerca. [9]

```
import nltk

# testo da analizzare
text = "Paris is the capital and largest city of France."

tokens = nltk.word_tokenize(text)
tagging = nltk.pos_tag(tokens)
named_entities = nltk.ne_chunk(tagging)
list(set([' '.join(w for w, t in elt) for elt in named_entities
          if type(elt) == nltk.tree.Tree]))

# Output: ['Paris ', 'France ']
```

5: Esempio di codice Python per l'estrazione di entità con *NLTK*.

### 3.6 Stanford CoreNLP

Stanford CoreNLP è un'implementazione Java di un *Named Entity Recognizer*, che può essere invocata in *Python* tramite la libreria *NLTK* (vedi 3.5). È stato creato nel 2014 da *Stanford NLP Group*, un team di docenti, programmatori e studenti dell'Università di Stanford che lavorano insieme su algoritmi che consentono ai computer di elaborare e comprendere i linguaggi umani.

*Stanford CoreNLP* fornisce una serie di strumenti di analisi del linguaggio naturale. Dato in input un testo in linguaggio naturale, supporta la lemmatizzazione, l'etichettatura delle parti del discorso, il riconoscimento di entità e molto altro. È stato originariamente sviluppato per l'inglese, ma ora fornisce anche diversi livelli di supporto per arabo, cinese, francese, tedesco e spagnolo. Non supporta (per adesso) la lingua italiana.

Il codice di *Stanford CoreNLP* è scritto in *Java* ed è concesso in licenza con *GNU General Public License v3*. [10]

```
import nltk
from nltk.tag import StanfordNERTagger

# testo da analizzare
text = "Paris is the capital and largest city of France."

# path del modello inglese e del jar di Stanford CoreNLP
model = './stanford/classifiers/english.muc.7class.distsim.crf.ser.gz'
jar = './stanford/stanford-ner.jar'

# Utilizza Stanford CoreNLP come tagger
tagger = StanfordNERTagger(model, jar)
```



```

tokens = nltk.word_tokenize(self.text)

chunk = []
entities = []
for token, tag in self.tagger.tag(tokens):
    if tag == 'O':
        # Concatena entità parziali adiacenti, se necessario
        if chunk:
            entities.append(' '.join([c[0] for c in chunk]))
            chunk = []
        else:
            chunk.append((token, tag))

list(set(entities))

# Output: ['Paris', 'France']

```

6: Esempio di codice Python per l'estrazione di entità con *Stanford CoreNLP*.

### 3.7 Stanza

Stanza è un *toolkit* di analisi del linguaggio naturale *Python*, introdotto nel 2020 da *Stanford NLP Group*, lo stesso team dietro alla libreria Java *Stanford CoreNLP* (vedi 3.6). [11]

Rispetto ai toolkit ampiamente utilizzati, *Stanza* presenta una pipeline completamente neurale indipendente dalla lingua per l'analisi del testo, e offre molte funzioni tra cui tokenizzazione, lemmatizzazione, etichettatura di parti del discorso e caratteristiche morfologiche, analisi delle dipendenze e riconoscimento di entità nominate.

*Stanza* è addestrato su un totale di 112 dataset di dati, e per il riconoscimento di entità supporta otto lingue: arabo, cinese, olandese, inglese, francese, tedesco, russo e spagnolo. Non supporta quindi per adesso la lingua italiana.

```

import stanza

# testo da analizzare
text = "Paris is the capital and largest city of France."

nlp = stanza.Pipeline('en') # carica modello inglese per l'analisi
doc = nlp(text)
list(set([e.text for e in doc.ents]))

# Output: ['Paris', 'France']

```

7: Esempio di codice Python per l'estrazione di entità con *Stanza*.

### 3.8 Tint

Tint (*The Italian NLP Tool*) è una pipeline open source basata su Java per l'elaborazione del linguaggio naturale esclusivamente in lingua italiana. [12]

Tint è basato su Stanford CoreNLP (vedi 3.6) e può essere utilizzato come strumento autonomo, incluso come libreria Java o come servizio API REST. È anche distribuito su Maven Central, quindi può essere facilmente integrato in un progetto esistente.

Il NER incluso in *Tint* si basa sullo stesso classificatore incluso in *Stanford CoreNLP* ed allenato sulla **Content Annotation Bank (I-CAB)** italiana, contenente circa 180.000

parole tratte dal quotidiano italiano *L'Adige*, e utilizzato per il compito di *Entity Recognition* di *Evalita 2009*. [13]

*Tint* implementa la maggior parte degli strumenti linguistici comuni, come la codifica delle parti del discorso, l'analisi delle dipendenze, la tokenizzazione e il riconoscimento di entità. In questo elaborato, *Tint* è stato utilizzato come servizio *API REST* (nel download è incluso uno script bash, *tint-server.sh*, per l'avvio del server locale) che è in grado di restituire i risultati dell'analisi in formato *JSON*.

```
import requests

# testo da analizzare
text = "Parigi e' la capitale della Francia."

# Il server Tint deve essere in esecuzione in locale nella porta 8012
base_url = 'http://localhost:8012/tint?text={}'.format(text)
json = requests.get(base_url).json()

entities = []
chunk = []
for sentence in json['sentences']:
    for token in sentence['tokens']:
        # entita': persone, luoghi o organizzazioni
        if token['ner'] in ['LOC', 'ORG', 'PER']:
            chunk.append(token)
        else:
            # Concatena entita' parziali adiacenti, se necessario
            if chunk:
                entities.append(' '.join([c['word'] for c in chunk]))
                chunk = []

# Output: entities = ['Parigi', 'Francia']
```

8: Esempio di codice Python per l'estrazione di entità con *Tint*.

### 3.9 Google Cloud

È possibile analizzare ed estrarre informazioni significative da testo non strutturato con le API di *Natural Language* messe a disposizione dalla piattaforma *Google Cloud*. [14]

*Natural Language* utilizza la tecnologia di machine learning tramite modelli pre-addestrati per fornire funzionalità come analisi del sentimento, analisi ed estrazione delle entità, classificazione di contenuti e analisi della sintassi.

Al momento per l'estrazione di entità sono supportate dieci lingue: cinese (semplificato e tradizionale), inglese, francese, tedesco, italiano, giapponese, coreano, portoghese, russo e spagnolo.

L'utilizzo dell'API *Natural Language* è a pagamento (pur offrendo 300\$ di credito ai nuovi iscritti), e il prezzo è calcolato mensilmente in base al numero di richieste effettuate. È richiesto un file *json* di autenticazione che può essere scaricato direttamente dall'interfaccia web.

```

from google.cloud import language_v1
from google.cloud.language_v1 import enums

# testo da analizzare
text = "Parigi e' la capitale della Francia."

# Autenticazione
client = language_v1.LanguageServiceClient
        .from_service_account_file('auth.json')

document = { "content": text, "type": enums.Document.Type.PLAIN_TEXT }
response = client.analyze_entities(document, enums.EncodingType.UTF8)

list(set([entity.mentions[0].text.content for e in response.entities
          if enums.Entity.Type(e.type).name in
              # entita': persone, luoghi o organizzazioni
              ['PERSON', 'LOCATION', 'ORGANIZATION']
          and enums.EntityMention.Type(e.mentions[0].type).name
              == 'PROPER']
        ))
)

# Output: ['Parigi', 'Francia']

```

9: Esempio di codice Python per l'estrazione di entità con *Google Cloud*.

## 4 Dataset scelti

Sono stati scelti due dataset già annotati con entità, uno in lingua inglese e uno in lingua italiana, descritti di seguito.

### 4.1 Dataset in lingua inglese

Il dataset inglese è stato scaricato da *Kaggle*, una comunità online di data scientist e professionisti dell'apprendimento automatico che consente, tra le altre cose, di condividere ed esplorare set di dati. [15]

Si tratta di un estratto del corpus **Groningen Meaning Bank (GMB)**, una risorsa semantica composta da oltre 10.000 testi in lingua inglese, tutti di pubblico dominio e disponibili gratuitamente per scopi di ricerca, già annotati appositamente per addestrare dei classificatori per la previsione di entità nominate.

Il file ha estensione *.csv* e pesa circa 15MB, ed è strutturato in questo modo:

- Il documento è suddiviso in frasi, identificate dalla colonna "*Sentence #*". Sono presenti oltre 45.000 frasi;
- Ogni riga rappresenta una parola della frase (ad esempio, la prima frase occupa 23 righe);
- La colonna *Word* contiene la parola della frase;
- La colonna *POS* indica la *Part of Speech* della parola, ovvero la classe morfologica (sostantivo, verbo, aggettivo, ecc..) a cui appartiene;

- La colonna *Tag* indica il *tag* associato alla parola, ovvero il tipo di entità che rappresenta: in questo caso siamo interessati ai tag *B-geo* (luogo), *B-per* (persona) e *B-org* (organizzazione). Se la parola non corrisponde a nessun tipo di entità viene indicato "O" come tag.

```

*** pd.read_csv('datasets/en/dataset_orig.csv', encoding='ISO-8859-1').head(24)
  Sentence #      Word POS  Tag
0 Sentence: 1  Thousands NNS   O
1         NaN      of   IN   O
2         NaN demonstrators NNS   O
3         NaN      have VBP   O
4         NaN    marched VBN   O
5         NaN    through   IN   O
6         NaN     London NNP B-geo
7         NaN      to    TO   O
8         NaN    protest   VB   O
9         NaN     the    DT   O
10        NaN     war    NN   O
11        NaN     in     IN   O
12        NaN     Iraq  NNP B-geo
13        NaN    and     CC   O
14        NaN   demand   VB   O
15        NaN     the    DT   O
16        NaN withdrawal NN   O
17        NaN      of     IN   O
18        NaN   British  JJ B-geo
19        NaN   troops  NNS   O
20        NaN     from   IN   O
21        NaN    that    DT   O
22        NaN   country  NN   O
23        NaN      .     .   O

```

Figura 2: Dataset inglese originale.

Le entità nominate sono annotate nel formato **IOB2**: il tag "B" (da "begin") indica il primo token dell'entità, mentre il tag "I" (da "inside") viene utilizzato per tutti gli altri token della stessa entità, se presenti (ad esempio, se l'entità è "Barack Obama", la parola "Barack" avrà token *B-per* e "Obama" avrà *I-per*).

La figura 2 mostra le prime 23 righe del dataset (corrispondenti alla prima frase del testo annotato - "Thousands of demonstrators have marched through London to protest the war in Iraq and demand the withdrawal of British troops from that country."), e la colonna *Tag* evidenzia correttamente la presenza di tre entità geografiche (*London*, *Iraq*, *British*).

#### 4.1.1 Preprocessamento dei dati

Può risultare scomodo caricare tutte le volte in memoria un dataset con centinaia di migliaia di righe: si è resa quindi necessaria una fase intermedia di preprocessamento dei dati:

- Sono state prese in considerazione solo le prime 25.000 righe del dataset originale;
- è stato scelto di associare una riga ad ogni frase del testo in modo da ridurre significativamente il numero di righe;
- La entità sono state raggruppate in un elenco unico per ciascuna frase;
- Sono state rimosse direttamente le frasi senza entità (inutili ai fini dell'elaborato).

Il risultato finale è esportato in un file *.csv* di soli 130kb contenente poco più di 1100 righe, e strutturato come mostrato in figura 3.

id	Sentence	Entities
1	Thousands of demonstrators have marched through London to protest the war in Iraq and demand the withdrawal of British troops from that country.	London, Iraq
2	Families of soldiers killed in the conflict joined the protesters who carried banners with such slogans as" Bush Number One Terrorist" and" Stop the Bombings."	Bush
3	They marched from the Houses of Parliament to a rally in Hyde Park.	Hyde Park
5	The protest comes on the eve of the annual conference of Britain's ruling Labor Party in the southern English seaside resort of Brighton.	Britain, Labor Party, Brighton
6	The party is divided over Britain's participation in the Iraq conflict and the continued deployment of 8,500 British troops in that country.	Iraq
7	The London march came ahead of anti-war protests today in other cities including Rome Paris and Madrid.	London, Rome, Paris, Madrid
8	The International Atomic Energy Agency is to hold second day of talks in Vienna Wednesday on how to respond to Iran's resumption of low-level uranium conversion.	International Atomic Energy Agency, Vienna
9	Iran this week restarted parts of the conversion process at its Isfahan nuclear plant.	Isfahan

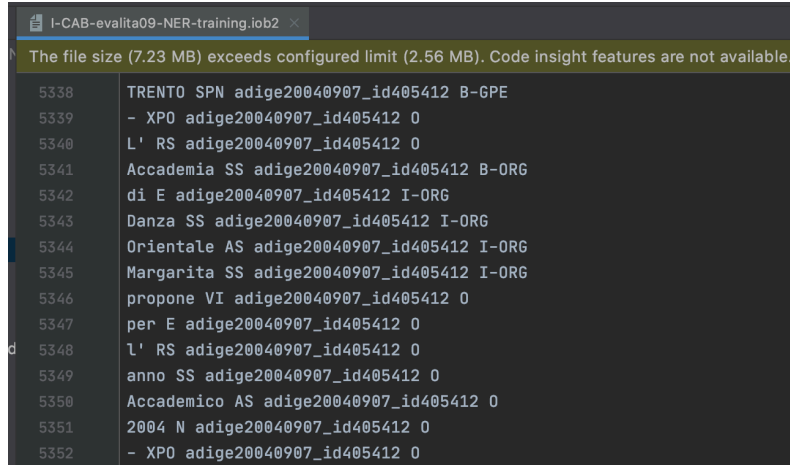
Figura 3: Dataset inglese finale.

## 4.2 Dataset in lingua italiana

Il dataset italiano è un estratto dell'**Italian Content Annotation Bank** (*I-CAB*), un corpus annotato composto da 525 notizie tratte dal quotidiano locale *L'Adige*, per un totale di circa 180.000 parole. La risorsa è liberamente disponibile per scopi di ricerca, previa accettazione di un contratto di licenza (e una attesa di due settimane), pertanto il modello risultante non può essere utilizzato per scopi commerciali. Il corpus è stato utilizzato, tra le altre cose, presso *EVALITA 2007* e *2009*, una campagna di valutazione periodica dello stato di avanzamento del *Natural Language Processing (NLP)* per la lingua italiana. [13]

Il file ha estensione *.iob2* e pesa 7,2 MB. È organizzato in questo modo:

- Il documento è suddiviso in oltre 500 brevi articoli;
- Ogni riga rappresenta una parola della frase;
- La prima colonna contiene la parola della frase;
- La seconda colonna indica la *Part of Speech* della parola, ovvero la classe morfologica (sostantivo, verbo, aggettivo, ecc..) a cui appartiene;
- La terza colonna indica l'*id* della notizia de *L'Adige* a cui appartiene l'articolo;
- La quarta colonna, infine, indica il *tag* associato alla parola, ovvero il tipo di entità che rappresenta: in questo caso siamo interessati ai tag *B-LOC* (luogo), *B-PER* (persona) e *B-ORG* (organizzazione). Se la parola non corrisponde a nessun tipo di entità viene indicato "O" come tag.



```

I-CAB-evalita09-NER-training.lob2 x
The file size (7.23 MB) exceeds configured limit (2.56 MB). Code insight features are not available.
5338 TRENTO SPN adige20040907_id405412 B-GPE
5339 - XPO adige20040907_id405412 0
5340 L' RS adige20040907_id405412 0
5341 Accademia SS adige20040907_id405412 B-ORG
5342 di E adige20040907_id405412 I-ORG
5343 Danza SS adige20040907_id405412 I-ORG
5344 Orientale AS adige20040907_id405412 I-ORG
5345 Margarita SS adige20040907_id405412 I-ORG
5346 propone VI adige20040907_id405412 0
5347 per E adige20040907_id405412 0
5348 l' RS adige20040907_id405412 0
5349 anno SS adige20040907_id405412 0
5350 Accademico AS adige20040907_id405412 0
5351 2004 N adige20040907_id405412 0
5352 - XPO adige20040907_id405412 0

```

Figura 4: Dataset italiano originale.

Anche per questo dataset le entità nominate sono annotate nel formato **IOB2** (vedi 4.1). La figura 4 mostra un esempio di articolo annotato ("*TRENTO - L'Accademia di Danza Orientale Margarita propone per l'anno Accademico 2004*") in cui sono individuate due entità (*TRENTO* e *Accademia di Danza Orientale Margarita*).

#### 4.2.1 Preprocessamento dei dati

Anche in questo caso è stata applicata una fase intermedia di preprocessamento dei dati:

- è stato scelto di associare una riga ad ogni frase dell'articolo, in modo da ridurre significativamente il numero di righe;
- Le entità sono state raggruppate in un elenco unico per ciascuna frase;
- Sono state rimosse direttamente gli articoli senza entità e gli articoli con meno di dieci parole.

Il risultato finale è esportato in un file *.csv* di circa 800kb contenente poco più di 10000 righe, e strutturato come mostrato in figura 5.

id	Sentence	Entities
241	TRENTO - L'Accademia di Danza Orientale Margarita propone per l'anno Accademico 2004 - 05 i suoi corsi di danza del ventre a cura della maestra e danzatrice Maria Rita Gandra a Riva Arco Torbole sul Garda a partire da quest'oggi con la prima lezione dimostrativa e gratuita con prenotazione telefonica o per e-mail.	Accademia di Danza Orientale Margarita, Maria Rita Gandra, Garda
247	Moratti vuole Egidì al suo fianco	Moratti, Egidì
249	Potrebbe prendere il posto di De Maio	De Maio
250	Non ha ancora abbandonato l'ufficio di via Belenzani - dove ha lavorato per 8 anni - e già fioccano le" offerte di lavoro".	via Belenzani
251	Dopo due mandati nella cabina di comando dell'Università di Trento dove si è fatto apprezzare ora non ha che l'imbarazzo della scelta.	Università di Trento
254	Massimo Egidì rettore uscente fra poche settimane lascerà il posto al neoletto Davide Bassi ma di sicuro non avrà problemi di" collocamento".	Davide Bassi
256	Nei giorni scorsi è stato dipinto come il candidato ideale alla presidenza dell'Irc futura Fondazione Kessler dopo l'" era Bonvicini".	Irc, Fondazione Kessler
257	Ma Egidì che ha avuto modo di allacciare forti legami a livello nazionale anche in quanto membro del direttivo della Crui (Conferenza dei rettori) potrebbe puntare ad un ruolo strategico a Roma a fianco del ministro Letizia Moratti con cui è in ottimi rapporti.	Egidì, Crui, Conferenza dei rettori, Letizia Moratti

Figura 5: Dataset italiano finale.

## 5 Esperimenti realizzati

Per una maggiore accuratezza gli esperimenti sono stati effettuati su 50 articoli in italiano e 50 in inglese. Ogni articolo è stato composto concatenando 10 frasi indipendenti tra loro. Ciascun articolo è stato quindi salvato in un file *.txt* all'interno della cartella del progetto (vedi figura 6).

```
Article #4

Devonish la Bailey e tutti i protagonisti gara per gara
UOMINI - 100 METRI / Quest'anno l'inglese Marlon Devonish oro olimpico con la 4x100 ha un record di 10" 32 ma il
personale è 10" 13 (del'98).
Il favorito è lo statunitense Joshua Johnson (10" 11 ma nel 2002 addirittura 9" 95).
In agguato Simone Collio (10" 20) lo sloveno Osovnikar (10" 15) e lo statunitense Streete - Thompson (10" 15).
Nella serie B occhio a Torrieri (10" 32) e Kaba Fantoni (10" 28).
400 METRI / Il giamaicano Blackwood è un finalista olimpico (8° personale di 44" 60).
Il keniano Sambu è un bel tipino (45" 33).
Andrea Barberi (45" 98) è il migliore dei nostri.
800 METRI / Il semifinalista olimpico Andrea Longo (1'44" 42 stagionale 1'43" 74 nel 2000) contro tutti;
anche contro il record italiano di Fiasconaro (1'43" 7 nel'73).
```

Figura 6: Esempio di articolo generato dal dataset italiano.

### 5.1 Valutazione della performance

Il primo esperimento realizzato ha lo scopo di valutare la performance di ciascun estrattore di entità. In lingua inglese sono stati valutati i seguenti tool: *Spacy* (3.4), *Polyglot* (3.3), *Natural Language* (3.2), *Dandelion* (3.1), *NLTK* (3.5), *Stanford CoreNLP* (3.6), *Google Cloud* (3.9) e *Stanza* (3.7).

Per l'italiano, invece, sono stati valutati ovviamente solo i tool che supportano la lingua: *Spacy* (3.4), *Polyglot* (3.3), *Natural Language* (3.2), *Dandelion* (3.1), *Tint* (3.8) e *Google Cloud* (3.9).

Per ciascun articolo (50 in inglese e 50 in italiano), sono state estratte le entità con ogni tool. A questo punto, conoscendo le entità "vere" per ogni articolo, è possibile definire i seguenti valori:

- Falsi negativi (FN): numero di entità vere non estratte;
- Falsi positivi (FP): numero di entità estratte che però non corrispondono a nessuna tra quelle "vere";
- Veri negativi (TN): sarebbe il numero di entità "false" non estratte, ma è sempre zero nel nostro caso;
- Veri positivi (TP): numero di entità vere estratte correttamente.

Actual Class	Predicted class		
		Class = Yes	Class = No
	Class = Yes	True Positive	False Negative
	Class = No	False Positive	True Negative

Figura 7: Esempio di matrice di confusione. (Fonte: *blog.exsilio.com*)

Da cui è possibile definire le seguenti metriche:

- **Accuracy:** è il rapporto tra il numero di entità corrette e il totale delle entità trovate ( $\frac{TP+TN}{TP+FP+FN+TN}$ );
- **Precision:** è il rapporto tra il numero di entità corrette e il totale delle entità estratte, includendo anche i falsi positivi ( $\frac{TP}{TP+FP}$ );
- **Recall:** è il rapporto tra il numero di entità corrette estratte e il totale delle entità presenti nell'insieme delle entità "vere" ( $\frac{TP}{TP+FN}$ );
- **F1 Score:** è la media armonica di precision e recall ( $2 \cdot \frac{Recall \cdot Precision}{Recall + Precision}$ ).

I risultati trovati per ciascun articolo (tabella di comparazione delle entità, metriche e matrici di confusione) sono stati salvati nella cartella del progetto (vedi figura 8, 9 e 10).

article\_4\_entities

	True Entities	Spacy Entities	Polyglot Entities	Dandelion Entities	NaturalLanguage Entities	Tint Entities	Google Cloud Entities
0	Fiasconaro	Marlon Devonish oro	Fiasconaro	Slovenia	Fiasconaro	Fiasconaro	Fiasconaro
1	Torrieri	Fiasconaro	Fantoni	Sambu	Thompson	Torrieri	italiano
2	Streete - Thompson	Thompson	statunitense Streete - Thompson	Andrea Barberi	Torrieri	Streete - Thompson	Streete - Thompson
3	serie B	Torrieri	Blackwood	Serie B	Streete	serie B	inglese
4	Sambu	Devonish la Bailey	statunitense	Stati Uniti d'America	giamaicano Blackwood	Sambu	giamaicano
5	Blackwood	Streete	Bailey	Marlon Devonish	B	Blackwood	Blackwood
6	Kaba Fantoni	serie B	Andrea Barberi	Andrea Longo (atleta)	keniano Sambu	Kaba Fantoni	Sambu
7	Osovnikar	Blackwood	Marlon Devonish	Simone Collio	Kaba Fantoni	Osovnikar	statunitense
8	Bailey	keniano Sambu	Andrea Longo		Osovnikar	Bailey	Kaba Fantoni
9	Joshua Johnson	Kaba Fantoni	statunitense Joshua Johnson		Bailey	Joshua Johnson	Osovnikar
10	Andrea Barberi	Osovnikar	Torrieri		Joshua Johnson	Andrea Barberi	sloveno
11	Marlon Devonish	UOMINI	Simone Collio		Andrea Barberi	Marlon Devonish	Bailey
12	Andrea Longo	Joshua Johnson			Marlon Devonish	Andrea Longo	Joshua Johnson
13	Devonish	Andrea Barberi			Andrea Longo	Devonish	Andrea Barberi
14	Simone Collio	Andrea Longo			Devonish	Simone Collio	Marlon Devonish
15		METRI			Simone Collio		Andrea Longo
16		Simone Collio					Torrieri
17							Simone Collio

Figura 8: Tabella di comparazione delle entità trovate nell'articolo di figura 6.

Matrici di confusione

Spacy

	Predetti Negativi	Predetti Positivi
Negativi	0	7
Positivi	0	10

Dandelion

	Predetti Negativi	Predetti Positivi
Negativi	0	5
Positivi	0	11

Polyglot

	Predetti Negativi	Predetti Positivi
Negativi	0	4
Positivi	3	8

Tint

	Predetti Negativi	Predetti Positivi
Negativi	0	0
Positivi	0	15

NL

	Predetti Negativi	Predetti Positivi
Negativi	0	5
Positivi	7	3

GCloud

	Predetti Negativi	Predetti Positivi
Negativi	0	5
Positivi	0	13

Figura 9: Matrici di confusione per ciascun tool relative all'articolo di figura 6.



Risultati				
	accuracy	recall	precision	f1
Spacy	0.666667	0.666667	0.588235	0.625
Polyglot	0.533333	0.533333	0.666667	0.592593
NL	0.2	0.2	0.375	0.26087
Dandelion	0.733333	0.733333	0.6875	0.709677
Tint	1	1	1	1
GCloud	0.866667	0.866667	0.722222	0.787879

Figura 10: Metriche calcolate per ciascun tool sull'articolo di figura 6..

I risultati per questo esperimento, ottenuti come la media calcolata su tutti gli articoli, sono illustrati nel capitolo 6.1.

## 5.2 Tempi di computazione

Il secondo esperimento realizzato ha lo scopo di valutare i tempi di computazione per ciascun tool. Sono stati quindi misurati i tempi di estrazione di entità di ciascun tool su 50 articoli diversi ed è stata poi fatta la media.

È importante sottolineare però che nel caso di *Dandelion* (3.1), *Google Cloud* (3.9) e *Tint* (3.8) questi risultati sono poco rilevanti, in quanto si tratta di tool che operano tramite API *REST* in cui il tempo di estrazione dipende soprattutto dai tempi di risposta del server.

Nel caso degli altri tool, è stato anche ignorato l'eventuale tempo necessario a caricare i modelli in memoria e, nel caso di *Tint*, il tempo di attesa per l'avvio del server locale.

I risultati ottenuti per questo esperimento sono illustrati nel capitolo 6.2.

## 5.3 Specifiche hardware

Tutti gli esperimenti sono stati effettuati su un *MacBook Pro (Retina, 13-inch, Mid 2014)* con processore 2,6 GHz Dual-Core Intel Core i5 e 8 GB di RAM (1600 MHz DDR3), con sistema operativo *macOS Catalina* (versione 10.15.6).

Sulla macchina è installato *Python* versione 3.7.6 e *Java* versione 13.0.2.

L'IDE utilizzato è *Pycharm (Professional Edition)* versione 2020.2.

## 6 Risultati ottenuti

### 6.1 Valutazione della performance

Di seguito è riportata la tabella della media delle metriche (accuracy, recall, precision e f1 score) calcolate per ciascun tool su 50 articoli in lingua **inglese**:

	accuracy	recall	precision	f1 score
Spacy	0.66701	0.66701	0.637997	0.649229
Polyglot	0.558444	0.558444	0.521301	0.53556
NL	0.705091	0.705091	0.682237	0.68868
Dandelion	0.545663	0.545663	0.525791	0.528783
NLTK	<b>0.723784</b>	<b>0.723784</b>	0.575887	0.637886
Stanford	0.655325	0.655325	<b>0.707102</b>	<b>0.674614</b>
GCloud	0.651893	0.651893	0.62159	0.632342
Stanza	0.684041	0.684041	0.645705	0.660724

Tabella 1: Tabella delle performance dei tool per l'estrazione di entità in lingua inglese.

Di seguito sono riportati gli stessi dati ma in forma di istogramma:

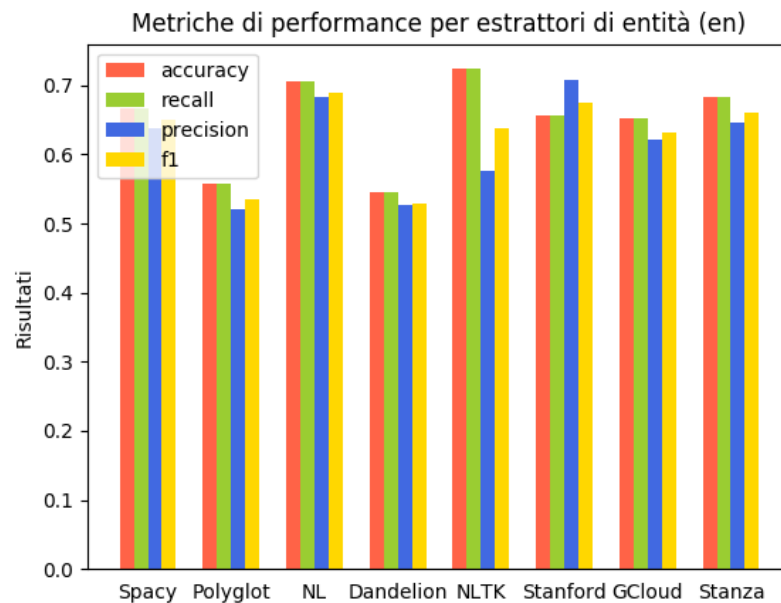


Figura 11: Istogramma relativo ai dati della tabella 1.

Di seguito invece è riportata la tabella della media delle metriche (accuracy, recall, precision e f1 score) calcolate per ciascun tool su 50 articoli in lingua **italiana**:

	accuracy	recall	precision	f1 score
<b>Spacy</b>	0.591253	0.591253	0.616677	0.595456
<b>Polyglot</b>	0.45896	0.45896	0.473719	0.46029
<b>NL</b>	0.517147	0.517147	0.585914	0.538017
<b>Dandelion</b>	0.287502	0.287502	0.496723	0.34487
<b>Tint</b>	<b>0.802747</b>	<b>0.802747</b>	<b>0.80021</b>	<b>0.79569</b>
<b>GCloud</b>	0.622233	0.622233	0.643944	0.625301

Tabella 2: Tabella delle performance dei tool per l'estrazione di entità in lingua italiana.

E in forma di istogramma:

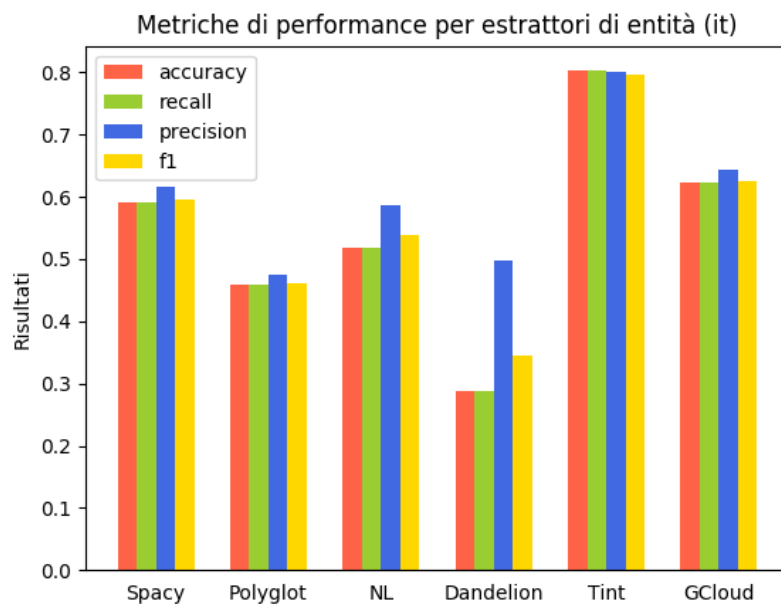


Figura 12: Istogramma relativo ai dati della tabella 2.

È importante però sottolineare che il risultato per *Tint* non è imparziale, in quanto il classificatore incluso in *Tint* è allenato sullo stesso dataset scelto per l'elaborato [13]. Questo porta sicuramente a problemi di sovradattamento (*overfitting*) e accuratezza più alta del normale.

Per quanto riguarda gli scarsi risultati di *Dandelion*, invece, c'è da dire che i valori ottenuti dipendono fortemente dalla soglia di confidenza scelta, e che le entità restituite dal servizio, oltre a luoghi, persone e organizzazioni, includono anche concetti astratti (es. "arte", "musica", "sport") che non è possibile escludere.

## 6.2 Tempi di computazione

Di seguito sono mostrate le medie dei tempi di estrazione di entità per ciascun tool. L'estrazione va considerata calcolata su un breve articolo di dieci frasi, sia in inglese che in italiano.

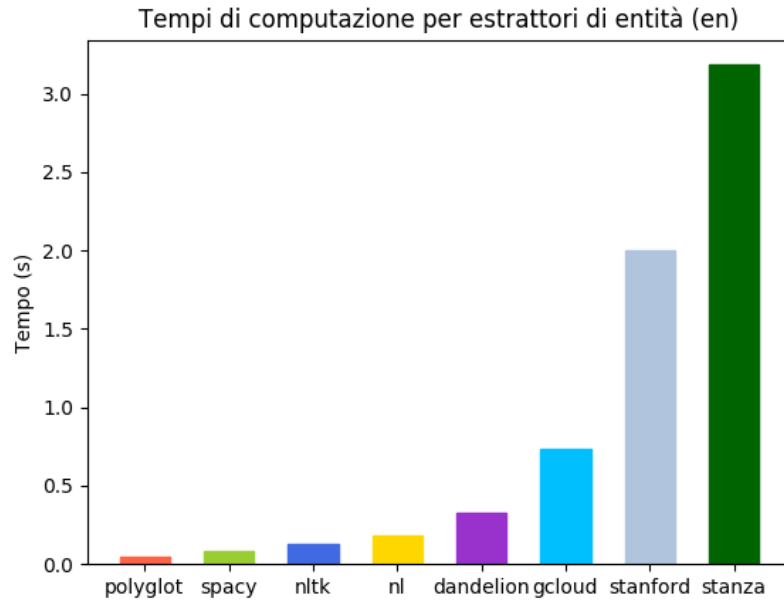


Figura 13: Tempi di estrazione di entità in lingua inglese.

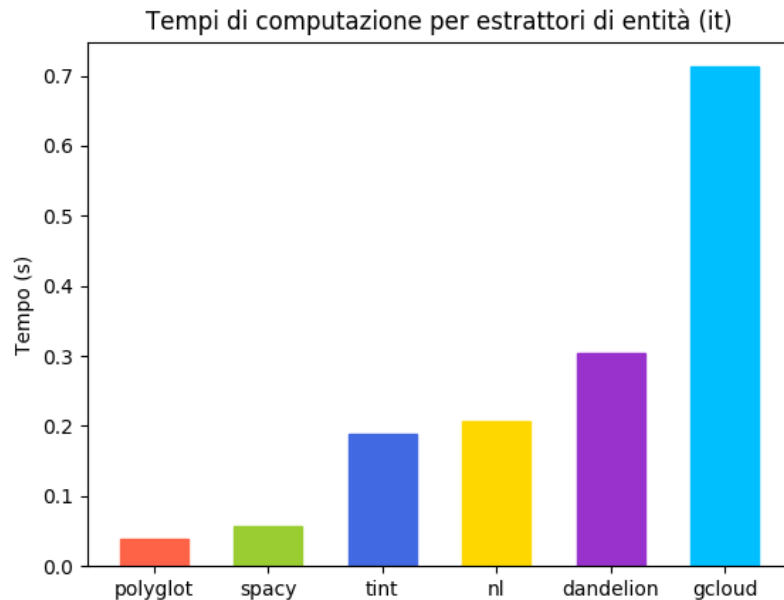


Figura 14: Tempi di estrazione di entità in lingua italiana.

Come già detto in precedenza, i tempi sono esclusivamente quelli di estrazione di entità: sono stati esclusi tempi di caricamento dei modelli o di attesa di avvio del server locale.

### 6.3 Conclusioni e sviluppi futuri

In generale, stabilire quale sia il tool migliore in base alle metriche (*accuracy*, *recall*, *precision*, *f1*) **dipende fortemente dallo scopo del progetto**: un estrattore con buona *accuracy* e *recall*, ma *precision* bassa (come nel caso di *NLTK* nel dataset inglese) è utile se lo scopo è quello di identificare tutte le possibili entità nel testo, a costo di trovarne anche altre "false". Viceversa, un estrattore con *precision* alta (come *Dandelion* nel dataset italiano) si adatta bene se si vuole trovare solo entità che hanno un'alta probabilità di verità, a costo di perdere alcune informazioni.

Ci sono in realtà dei problemi legati alla misura della qualità dell'output di un estrattore di entità. Uno di questi è legato alla presenza di **match parziali**, che non dovrebbero essere considerati nè come successi e nè come fallimenti completi. Ad esempio, se una frase cita il "Parco Nazionale dello Stelvio", è possibile che un estrattore in output restituisca soltanto l'entità "Stelvio" (comune di Bolzano): per le metriche definite prima, questo risultato viene considerato totalmente sbagliato nonostante abbia comunque a che fare con l'entità originale.

Si potrebbe pensare semplicemente di contare la frazione di token identificata correttamente all'interno dell'entità (l'estrazione di "Stelvio" al posto di "Parco Nazionale dello Stelvio" in questo caso conterebbe come  $1/4 = 0.25$ , invece di zero).

Questo può però portare ad un'altra serie di problemi, uno tra tutti dato dal fatto che non è detto che parte di un'entità sia legato all'entità stessa (ad esempio, l'entità "Istituto Agrario di San Michele" in provincia di Trento non ha niente a che fare con "San Michele" arcangelo).

Per quanto riguarda i possibili sviluppi futuri, potrebbe essere interessante valutare i tool anche in base al *tipo* di entità estratta (persona, organizzazione o luogo) e verificare che coincida con quello vero. Un altro sviluppo potrebbe essere quello di occuparsi, oltre al riconoscimento di entità, anche di **Named Entity Linking** (*NEL*), ovvero quella parte di elaborazione del linguaggio naturale che si occupa di assegnare un'identità univoca (come ad esempio un *URI*) e disambiguare le entità trovate.

## Riferimenti bibliografici

- [1] Wikipedia, “Natural language processing — Wikipedia, the free encyclopedia.” <http://en.wikipedia.org/w/index.php?title=Natural\%20language\%20processing&oldid=888277178>, 2019.
- [2] Wikipedia, “Named-entity recognition — Wikipedia, the free encyclopedia.” <http://en.wikipedia.org/w/index.php?title=Named-entity\%20recognition&oldid=882339931>, 2019.
- [3] A. Zamberletti, “Named entity extraction: l’approccio machine learning nella piattaforma gate/annie,” 2017. <https://amslaurea.unibo.it/13328/>.
- [4] Dandelion, “Semantic text analytics as a service.” <https://dandelion.eu>.
- [5] Apple, “Natural language - analyze natural language text and deduce its language-specific metadata..” <https://developer.apple.com/documentation/naturallanguage>.
- [6] Rami Al-Rfou, “Polyglot - a natural language pipeline that supports massive multilingual applications..” <https://polyglot.readthedocs.io/en/latest/>.
- [7] Spacy, “Industrial-strength natural language processing.” <https://spacy.io>.
- [8] NLTK Project, “Natural language toolkit - a leading platform for building python programs to work with human language data.” <http://www.nltk.org>.
- [9] Wikipedia, “Natural Language Toolkit — Wikipedia, the free encyclopedia.” [https://it.wikipedia.org/wiki/Natural\\_Language\\_Toolkit](https://it.wikipedia.org/wiki/Natural_Language_Toolkit), 2019.
- [10] Stanford NLP Group, “Corenlp - your one stop shop for natural language processing in java.” <https://stanfordnlp.github.io/CoreNLP/>.
- [11] P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning, “Stanza: A Python natural language processing toolkit for many human languages,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2020.
- [12] A. Palmero Aprosio and G. Moretti, “Tint – a java-based pipeline for natural language processing (nlp) in italian..” <http://tint.fbk.eu>.
- [13] FBK | Fondazione Bruno Kessler, “Italian content annotation bank (i-cab).” <http://ontotext.fbk.eu/icab.html>.
- [14] Google, “Natural language - derive insights from unstructured text using google machine learning..” <https://cloud.google.com/natural-language>.
- [15] Abhinav Walia, “Annotated corpus for named entity recognition.” [https://www.kaggle.com/abhinavwalia95/entity-annotated-corpus?select=ner\\_dataset.csv](https://www.kaggle.com/abhinavwalia95/entity-annotated-corpus?select=ner_dataset.csv).