

# NATURAL LANGUAGE PROCESSING

# المعالجة اللغوية الطبيعية



# المحتويات

				التطبيقات	العقبات و التحديات	تاريخ NLP	ما هو NLP	المحتويات	1) مقدمة
					البحث في النصوص	ملفات pdf	الملفات النصية	المكتبات	2) أساسيات NLP
T.Visualization	Syntactic Struc.	Matchers	Stopwords	NER	Stem & Lemm	POS	Sent. Segm.	Tokenization	3) أدوات NLP
	Dist. Similarity	Text Similarity	TF-IDF	BOW	Word2Vec	T. Vectors	Word embed	Word Meaning	4) المعالجة البسيطة
T. Generation	NGrams	Lexicons	GloVe	L. Modeling	NMF	LDA	T. Clustering	T. Classification	5) المعالجة المتقدمة
	Summarization & Snippets		Ans. Questions		Auto Correct	Vader	Naïve Bayes	Sent. Analysis	
Search Engine	Relative Extraction		Information Retrieval		Information Extraction		Data Scraping	Tweet Collecting	6) تجميع البيانات
					Rec NN\TNN	GRU	LSTM	Seq to Seq	7) RNN
Chat Bot	Gensim	FastText	Bert	Transformer	Attention Model	T. Forcing	CNN	Word Cloud	8) تكتيكات حديثة

## القسم الثالث : أدوات NLP

### الجزء الخامس : Named-Entity Recognition

=====

هي اختصار Name Entity Recognition وهي الخاصة بالتعرف علي و تصنيف كلمات هامة , مثل اسماء الاشخاص , او المؤسسات او اسماء الدول او المدن , الوقت , الاموال , النسب المئوية , وكذلك يقوم بتحديد اسماء الاعلام من اسماء الاشخاص و الشركات و المدن و العملات , هكذا

A very important sub-task: **find** and **classify** names in text, for example:

- The decision by the independent MP **Andrew Wilkie** to withdraw his support for the minority **Labor** government sounded dramatic but it should not further threaten its stability. When, after the **2010** election, **Wilkie**, **Rob Oakeshott**, **Tony Windsor** and the **Greens** agreed to support **Labor**, they gave just two guarantees: confidence and supply.

Person  
Date  
Location  
Organi-  
zation



و هنا تطبيق بسيط لها , فجملة مثل :

Foreign minister spokesman Shen Guofang told Reuters . .

سيتم عملها كالتالي , مع العلم ان حرف O يشير الي other اي كلمة اخري

Foreign	ORG	}	Standard evaluation is per entity, <i>not</i> per token
Ministry	ORG		
spokesman	O		
Shen	PER		
Guofang	PER		
told	O		
Reuters	ORG		
:	:		

## و من تطبيقات NER :

- يمكن ربط الاسماء المستخرجة بروابط لها ( ويكيبيديا )
- يتم ربط منتجات شركات معينة ( سيارة civic يتم ربطها بـ Honda )
- يتم ربط اسئلة معينة بإجابات لها في مكان آخر

\* \* \* \* \*

## كيف يتم تدريب الموديل لينجح في عمل NER للكلمات ؟

يتم هذا عبر خطوات عدة :

- تجميع كمية كبيرة و كافية للبيانات
- يدويا عمل تحديد لنوع كل كلمة من الكلمات
- عمل آلية مناسبة لاستخراج ال features المطلوبة من الكلمات
- عمل خوارزم ML لتدريبه علي الفيتشرز و تحديد الفئات
- ثم عمل الاختبار , لحساب مدي كفاءته في تحديد الفئات

و هنا مثال لجمله عامة :

Fred showed Sue Mengqui Huang`s new painting

فتم تحديد ان هناك كلمات هي اسماء Per و كلمات أخرى O

و يتم هذا عبر ما يسمى IO Encoding و هي اختصار Inside Outside Encoding , والتي نقوم فيها بتحديد فئة كل كلمة

وهناك خصوصية في الاسماء , فبعض الاسماء تكون اسم واحد لشخص معين Fred و بعض الاسماء تكون اسمين متتالين لنفس الشخص Mengqui Huang , فال IO Encoding لم تقم بالتفريق بين هذا و هذا

لذا أحيانا يتم استخدام IOB Encoding و ال B هنا لكلمة begin , اي اننا نحدد هل هذا الاسم هو بداية اسم طويل ام لا

فنري في كلمات (Fred Sue Mengqui) انها B , لان كل منها هي بداية اسم , بينما Huang ليست كذلك لانها ليست بداية اسم

علي أن هذا الامر يستهلك وقتها و ذاكرة اكبر , و فائدته اقل

	IO encoding	IOB encoding
Fred	PER	B-PER
showed	O	O
Sue	PER	B-PER
Mengqiu	PER	B-PER
Huang	PER	I-PER
's	O	O
new	O	O
painting	O	O

\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*

ومن المهم تحديد الفيتشرز التي سيتم استخدامها في التدريب . .

## فیشکل اساسی هی :

- الكلمة الحالية
- الكلمة السابقة او التالية او كلاهما
- قيمة POS للكلمة

\* \* \* \* \*

و احيانا يكون هناك تأثير لمتلازمة حروف معينة , فقم تم عمل تجربة للبحث عن حروف معينة في قلب 5 فئات من الكلمات هي ( الأدوية , الشركات , الأفلام , الأماكن , الأشخاص )

و كانت النتائج كالتالي , وهي ما توضح مدي تأثير تواجد حروف معينة في تحديد نوع الكلمة

و نفس الأمر فى اللغة العربية , فكلمة "كفر , او نجع" ترتبط بالمكان , كلمة "ابو , ام" بالاسماء , كلمة "ابن" بالشتائم ,





و يتم اظهار قيم NER عبر استخدام الكلاس ents بعد doc , بعد ان نقوم بعمل nlp() لها , مثل :

```
doc = nlp(u'Apple to build a Hong Kong factory for $6 million')
```

```
for ent in doc.ents:  
    print(ent.label_)
```

و هناك عدد من الدوال في ents , وهي :

```
print([ent for ent in doc.ents])
```

<code>`ent.text`</code>	The original entity text
<code>`ent.label`</code>	The entity type's hash value
<code>`ent.label_`</code>	The entity type's string description
<code>`ent.start`</code>	The token span's <i>*start*</i> index position in the Doc
<code>`ent.end`</code>	The token span's <i>*stop*</i> index position in the Doc
<code>`ent.start_char`</code>	The entity text's <i>*start*</i> index position in the Doc
<code>`ent.end_char`</code>	The entity text's <i>*stop*</i> index position in the Doc

## وايضاً تفسير الرموز الخارجة منها :

TYPE	DESCRIPTION	EXAMPLE
`PERSON`	People, including fictional.	*Fred Flintstone*
`NORP`	Nationalities or religious or political groups.	*The Republican Party*
`FAC`	Buildings, airports, highways, bridges, etc.	*Logan International Airport, The Golden Gate*
`ORG`	Companies, agencies, institutions, etc.	*Microsoft, FBI, MIT*
`GPE`	Countries, cities, states.	*France, UAR, Chicago, Idaho*
`LOC`	Non-GPE locations, mountain ranges, bodies of water.	*Europe, Nile River, Midwest*
`PRODUCT`	Objects, vehicles, foods, etc. (Not services.)	*Formula 1*
`EVENT`	Named hurricanes, battles, wars, sports events, etc.	*Olympic Games*
`WORK_OF_ART`	Titles of books, songs, etc.	*The Mona Lisa*
`LAW`	Named documents made into laws.	*Roe v. Wade*
`LANGUAGE`	Any named language.	*English*
`DATE`	Absolute or relative dates or periods.	*20 July 1969*
`TIME`	Times smaller than a day.	*Four hours*
`PERCENT`	Percentage, including "%".	*Eighty percent*
`MONEY`	Monetary values, including unit.	*Twenty Cents*
`QUANTITY`	Measurements, as of weight or distance.	*Several kilometers, 55kg*
`ORDINAL`	"first", "second", etc.	*9th, Ninth*
`CARDINAL`	Numerals that do not fall under another type.	*2, Two, Fifty-two*



نتناول هنا الكود المستخدم :

فلو كان لدينا جملة هكذا :

```
import spacy
nlp = spacy.load('en_core_web_sm')
doc1 = nlp(u'Apple to build a Hong Kong factory for $6 million')
```

فلو تم اظهار التوكنس ستكون هكذا :

```
for token in doc1:
    print(token.text, end=' | ')
```

اما لو تم استخدام الميثود `ents` فستقوم المكتبة باختيار الكلمات المهمة هما و هي ( Apple , Hong Kong , 6 Million ) بل و ستقوم ايضا بمعرفة المعني التقريبي لكل كلمة و تصنيفها و معرفة مكانها

```
for ent in doc1.ents:
    print(ent.text)
```



```
print(ent.label)
print(ent.label_)
print(str(spacy.explain(ent.label_)))
print(ent.start)
print(ent.end)
print(ent.start_char)
print(ent.end_char)
print('-----')
```

و يمكن عمل دالة تقوم باظهار كل التفاصيل عن الـ entities في الجملة هكذا

```
def show_ents(doc):
    if doc.ents:
        for ent in doc.ents:
            print(ent.text+' - '+ent.label_+' - '+str(spacy.explain(ent.label_)))
            print('-----')
    else:
        print('No named entities found.')
```

فاذا طبقناها علي جملة عادية ليس فيها entities ستكون

```
show_ents(nlp('Hi how are you'))
```

اما اذا استخدمناها علي جملة فيها entities

```
show_ents(nlp('May I go to Washington, DC next May to see the Washington Monument?'))
```

فسيقوم بعرض كل التفاصيل , مع ملاحظة ان spacy كان ذكيا لدرجة التفريق بين may في اول الكلام وهي سؤال و بالتالي ليست كلمة هامة , وبين May في وسط الكلام و هي شهر , كذلك بين washington المدينة في وسط الكلام , ونفس اللفظ كمتحف في نهاية الكلام

و هنا مثال اخر

```
show_ents(nlp('Can I please borrow 500 dollars from you to buy some Microsoft stock?'))
```

\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\_\*\*

و في حالة كان لدينا اسم هام و غير موجود في الـ entities فيمكن اضافته بخطوات محددة

فلو كان لدينا جملة :

```
show_ents(nlp('CPRO to build a U.K. factory for $6 million'))
```

فنجد ان اسم شركة CPRO لم يتمكن سباسي من تمييز , فيمكن اضافته هكذا :

يتم استدعاء الجزء الخاص بالمنظمات ORG

```
from spacy.tokens import Span
doc = nlp('CPRO to build a U.K. factory for $6 million')
```

ثم نقول لسباسي ان في الملف doc امسك التوكن 0 الي 1 قم بعمل ent جديد باسم new\_ent ثم نقول له اصف هذا الـ new\_ent الي الـ ent الخاصة بالملف

```
ORG = doc.vocab.strings[u'ORG']
```

```
new_ent = Span(doc, 0, 1, label=ORG)
```

```
doc = nlp('CPRO to build a U.K. factory for $6 million')
doc.ents = list(doc.ents) + [new_ent]
```

و هنا بعرض الـ ent يظهرها

```
show_ents(doc)
```

و كالمعتاد , اذا ما تم اغلاق الكرنيل فلام يتم حفظها في اصل سباسي

\* \* \* \* \*

هنا نستخدم noun\_chunks وهي الاسماء المميزة , فمثلا هنا سيقوم باختيار جمل معينة

```
doc2 = nlp(u"Autonomous cars shift insurance liability toward manufacturers.")
```

```
for chunk in doc2.noun_chunks:
    print(chunk.text)
    print(chunk.root.text)
    print(chunk.root.dep_)
    print(spacy.explain(chunk.root.dep_))
    print(chunk.label)
    print(chunk.label_)
    print(spacy.explain(chunk.label_))
    print(chunk.start)
    print(chunk.end)
    print(chunk.start_char)
    print(chunk.end_char)
    print('-----')
```



