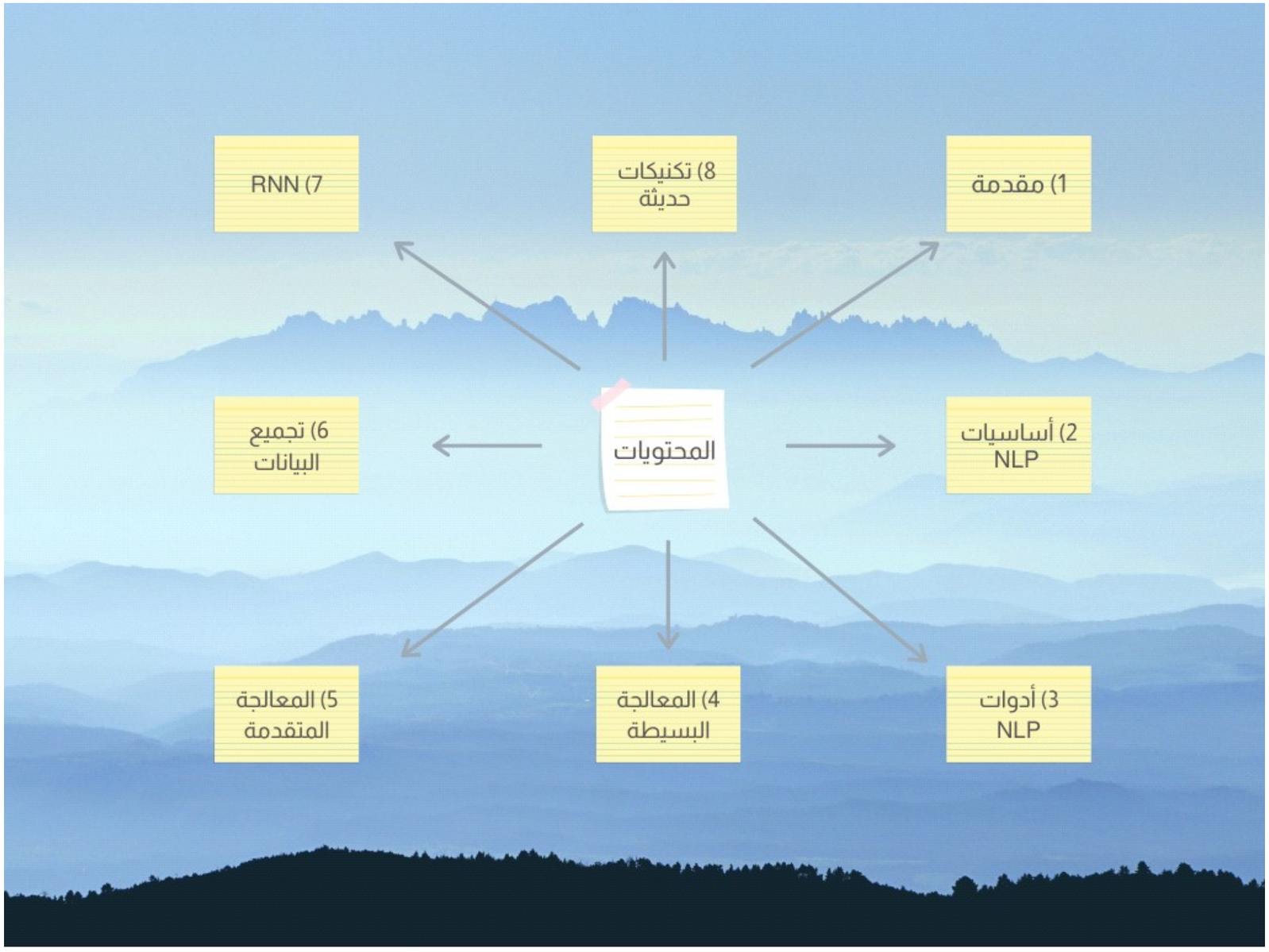
Natural Language Processing المعالجة اللغوية الطبيعية



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)المعاجلة 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	RNN (7
Chat Bot	Gensim	FastText	Bert	Transformer	Attention Model	T. Forcing	CNN	Word Cloud	8)تكنيكات حديثة

القسم الثالث : أدوات NLP

الجزء الثالث : POS

==========

الأجزاء من الخطاب Part of Speech

نتحدث الآن عن POS وهو الخاص بتحديد نوع الكلمة نحويا , هل هي فعل ام اسم او صفة , بناء علي سياق الكلمة الجملة التي فيها , و ليس الكلمة نفسها

فهي معتمدة علي ان معني اي كلمة ليس في ذاتها ولكن في مضمونها و سياقها و حسب الكلمات المحيطة بيها , و بالتالي تقوم بعمل تفسير لكل كلمة حسب محتواها و مضمونها و سياقها , وتصنيفها وسط اقسام كتير

فكلمة (ذهب) او play لها العديد من الاستخدامات و المعاني , بناء علي السياق المستخدم

و أول تحديد لأنواع الكلمات كان علي يد أرسطو في القرن الرابع قبل الميلاد . لكن التقسيم الأكثر دقة كان علي يد ثراكس في القرن الأول قبل الميلاد , والذي قام بتحديد 8 اصناف للكلمات , وهي قريبة من التصنيف الحالي

Perhaps starting with Aristotle in the West (384–322 BCE), there was the idea of having parts of speech

- a.k.a lexical categories, word classes, “tags”, POS

It comes from Dionysius Thrax of Alexandria (c. 100 BCE) the idea that is still with us that there are 8 parts of speech

- But actually his 8 aren’t exactly the ones we are taught today
- Thrax: noun, verb, article, adverb, preposition, conjunction, participle, pronoun
- School grammar: noun, verb, adjective, adverb, preposition, conjunction, pronoun, interjection

_

و هناك ما يسمي الفئات المفتوحة و المغلقة للكلمات

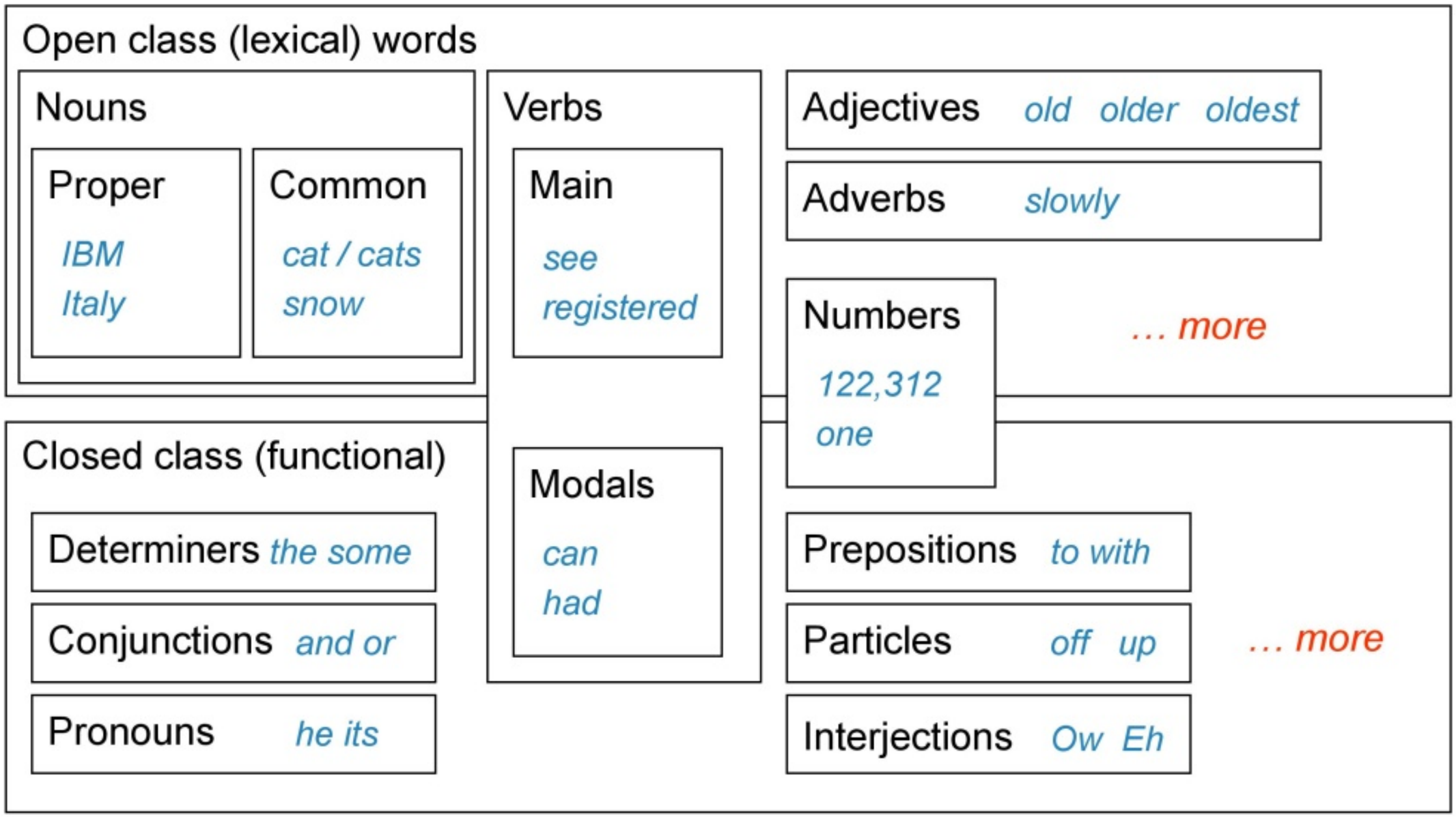
الفئات المغلقة هي للكلمات التي يمكن حصرها و ان نعرفها , ونادرا ما يتم اضافة شئ جديد عليها , مثل المحددات a , an , the , و هكذا .
بينما المفتوحة هي للكلمات التي لا يمكن حصرها و يمكن اضافة المزيد عليها , مثل الافعال او الاسماء

Open vs. Closed classes

- Closed:
  - determiners: ***a, an, the***
  - pronouns: ***she, he, I***
  - prepositions: ***on, under, over, near, by, …***
  - Why "closed"?
- Open:
  - Nouns, Verbs, Adjectives, Adverbs.

و التقسيم العام للكلمات كالتالي :

- الفئات المفتوحة :
  - الأسماء :
    - المخصصة (محددة بشئ معين): اسم شركة , دولة , مدينة
    - العامة : كلب , قطة
  - الأفعال :
    - الأفعال الأساسية
  - الصفات
  - الحال
- الفئات المغلقة :
  - المحددات : a , the
  - الروابط : and , or
  - الضمائر و ضمائر الملكية : he , it , his
  - حروف الجر : to , with
  - اللواحق off , up
  - الإضافات : oh , eh , wow
- الأرقام : و هي تتراوح بين المفتوح و المغلق


Open class (lexical) words
Nouns
Proper
IBM
Italy
Common
cat / cats
snow
Verbs
Main
see
registered
Modals
can
had
Adjectives old older oldest
Adverbs slowly
Numbers
122,312
one
… more
Closed class (functional)
Determiners the some
Conjunctions and or
Pronouns he its
Prepositions to with
Particles off up
Interjections Ow Eh
… more

_

علي انه يجب الانتباه إلي ان العديد من الكلمات لها الكثير من المسميات , بناء علي موضعها في الجملة و سياقها و الكلمات المحيطة

Words often have more than one POS: *back*

- The *back* door = JJ
- On my *back* = NN
- Win the voters *back* = RB
- Promised to *back* the bill = VB

The POS tagging problem is to determine the POS tag for a particular instance of a word.

ايضا هنا كلمة around

- Mrs/NNP Shaefer/NNP never/RB got/VBD around/RP to/TO joining/VBG

- All/DT we/PRP gotta/VBN do/VB is/VBZ go/VB around/IN the/DT corner/NN

- Chateau/NNP Petrus/NNP costs/VBZ around/RB 250/CD

و هنا جملة , نري ان كلمتي plays , well لهما معاني كثيرة , لكن يتم تحديدها هنا بناء علي موضعها

- Input: Plays well with others
- Ambiguity: NNS/VBZ UH/JJ/NN/RB IN NNS
- Output: Plays/VBZ well/RB with/IN others/NNS

Penn Treebank POS tags

- Uses:
  - Text-to-speech (how do we pronounce “lead”?)
  - Can write regexps like (Det) Adj* N+ over the output for phrases, etc.
  - As input to or to speed up a full parser
  - If you know the tag, you can back off to it in other tasks

ماذا عن دقتها ؟

How many tags are correct? (Tag accuracy)

- About 97% currently
- But baseline is already 90%
  - Baseline is performance of stupidest possible method
    - Tag every word with its most frequent tag
    - Tag unknown words as nouns
- Partly easy because
  - Many words are unambiguous
  - You get points for them (*the*, *a*, etc.) and for punctuation marks!

و أيضا التحديات في هذا الأمر

About 11% of the word types in the Brown corpus are ambiguous with regard to part of speech

But they tend to be very common words. E.g., *that*

- I know *that* he is honest = IN
- Yes, *that* play was nice = DT
- You can't go *that* far = RB

40% of the word tokens are ambiguous

_

و هناك العديد من مصادر المعلومات في سياق الجملة لتحديد الـ POS

مثل : الكلمات القريبة من الكلمة نفسها , ففي هذه الجملة , لا يمكن تحديد معني الكلمات دون النظر في الكلمات المجاورة

What are the main sources of information for POS tagging?

- Knowledge of neighboring words
  - Bill saw that man yesterday
  - NNP NN DT NN NN
  - VB VB(D) IN VB NN
- Knowledge of word probabilities
  - *man* is rarely used as a verb….

The latter proves the most useful, but the former also helps

كما ان هناك عدد من المعلومات في الكلمة نفسها مثل :
شكلها , كابيتال ام سمول , هل اول حرف كابيتال ؟ , الحروف السابقة لها , الحروف اللاحقة لها , وجود ارقام جوارها

Can do surprisingly well just looking at a word by itself:

- Word — the: the → DT
- Lowercased word — Importantly: importantly → RB
- Prefixes — unfathomable: un- → JJ
- Suffixes — Importantly: -ly → RB
- Capitalization — Meridian: CAP → NNP
- Word shapes — 35-year: d-x → JJ

Then build a maxent (or whatever) model to predict tag

- Maxent P(t|w): — 93.7% overall / 82.6% unknown

و هنا كفاءة عدد من الخوازرميات في اكتشاف الPOS

Rough accuracies:

• Most freq tag:	~90% / ~50%
• Trigram HMM:	~95% / ~55%
• Maxent P(t|w):	93.7% / 82.6%
• TnT (HMM++):	96.2% / 86.0%
• MEMM tagger:	96.9% / 86.9%
• Bidirectional dependencies:	97.2% / 90.0%
• Upper bound:	~98% (human agreement)

Most errors on unknown words

و لزيادة الدقة علينا ان :

- مراعاة الكلمات القريبة منها
- ليس لان اول حرف capital فهي تعتبر noun فقد تكون بداية الجملة

Build better features!

			RB				
PRP	VBD	IN	RB	IN	PRP	VBD	.
They	left	as	soon	as	he	arrived	.

- We could fix this with a feature that looked at the next word

	JJ				
NNP	NNS	VBD		VBN	.
Intrinsic	flaws	remained		undetected	.

- We could fix this by linking capitalized words to their lowercase versions

_

نبدأ بتحميل الملفات الخاصة بها ,

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

و نقوم بعرض كلا من pos , dep , tag و تفسير لكلا منهم

```
doc1 = nlp('''The son of a salesman who later operated an electrochemical factory, instein was born in the German Empire, but moved to Switzerland in 1895 and renounced his German citizenship in 1896. Specializing in physics and mathematics, he received his academic teaching diploma from the Swiss Federal Polytechnic School (German: eidgenössische polytechnische Schule) in Zürich in 1900. The following year, he acquired Swiss citizenship, which he kept for his entire life. After initially struggling to find work, from 1902 to 1909 he was employed as a patent examiner at the Swiss Patent Office in Bern.''')

for token in doc1:
    print('Words is   : ' , token.text)
    print('POS is   : ' , token.pos ,'===',token.pos_  , '===', spacy.explain(token.pos_))
    print('Dep is   : ' , token.dep , '===',token.dep_, '===', spacy.explain(token.dep_))
    print('Tag is   : ' , token.tag , '===',token.tag_, '===', spacy.explain(token.tag_))
    print('----------------------')
```

مع ملاحظة ان الاتريبيوت _pos و ياتي بتوع الكلمة وهو VERB , وهي لها العديد من القيم , تتلخص هنا :

POS	DESCRIPTION	EXAMPLES
ADJ	adjective	*big, old, green, incomprehensible, first*
ADP	adposition	*in, to, during*
ADV	adverb	*very, tomorrow, down, where, there*
AUX	auxiliary	*is, has (done), will (do), should (do)*
CONJ	conjunction	*and, or, but*
CCONJ	coordinating conjunction	*and, or, but*
DET	determiner	*a, an, the*
INTJ	interjection	*psst, ouch, bravo, hello*
NOUN	noun	*girl, cat, tree, air, beauty*
NUM	numeral	*1, 2017, one, seventy-seven, IV, MMXIV*
PART	particle	*'s, not,*
PRON	pronoun	*I, you, he, she, myself, themselves, somebody*
PROPN	proper noun	*Mary, John, London, NATO, HBO*
PUNCT	punctuation	*., (,), ?*
SCONJ	subordinating conjunction	*if, while, that*
SYM	symbol	*$, %, §, ©, +, −, ×, ÷, =, :), 😝*
VERB	verb	*run, runs, running, eat, ate, eating*
X	other	*sfpksdpsxmsa*
SPACE	space	

بينما الاتريبيوت _tag ياتي بتفاصيل كاملة وهي هنا : VBD و التي تم تفسيرها عبر استخدام spacy.explain و التي تعني verb past tense و باقي التفسيرات هنا

POS	Description	Fine-grained Tag	Description	Morphology
ADJ	adjective	AFX	affix	Hyph=yes
ADJ		JJ	adjective	Degree=pos
ADJ		JJR	adjective, comparative	Degree=comp
ADJ		JJS	adjective, superlative	Degree=sup
ADJ		PDT	predeterminer	AdjType=pdt PronType=prn
ADJ		PRP$	pronoun, possessive	PronType=prs Poss=yes
ADJ		WDT	wh-determiner	PronType=int rel
ADJ		WP$	wh-pronoun, possessive	Poss=yes PronType=int rel
ADP	adposition	IN	conjunction, subordinating or preposition	
ADV	adverb	EX	existential there	AdvType=ex
ADV		RB	adverb	Degree=pos
ADV		RBR	adverb, comparative	Degree=comp
ADV		RBS	adverb, superlative	Degree=sup
ADV		WRB	wh-adverb	PronType=int rel
CONJ	conjunction	CC	conjunction, coordinating	ConjType=coor
DET	determiner	DT	determiner	
INTJ	interjection	UH	interjection	
NOUN	noun	NN	noun, singular or mass	Number=sing
NOUN		NNS	noun, plural	Number=plur

NOUN		NNS	noun, plural	Number=plur
NOUN		WP	wh-pronoun, personal	PronType=int rel
NUM	numeral	CD	cardinal number	NumType=card
PART	particle	POS	possessive ending	Poss=yes
PART		RP	adverb, particle	
PART		TO	infinitival to	PartType=inf VerbForm=inf
PRON	pronoun	PRP	pronoun, personal	PronType=prs
PROPN	proper noun	NNP	noun, proper singular	NounType=prop Number=sign
PROPN		NNPS	noun, proper plural	NounType=prop Number=plur
PUNCT	punctuation	-LRB-	left round bracket	PunctType=brck PunctSide=ini
PUNCT		-RRB-	right round bracket	PunctType=brck PunctSide=fin
PUNCT	,	,	punctuation mark, comma	PunctType=comm
PUNCT	:	:	punctuation mark, colon or ellipsis	
PUNCT	.	.	punctuation mark, sentence closer	PunctType=peri
PUNCT	"	"	closing quotation mark	PunctType=quot PunctSide=fin
PUNCT	""	""	closing quotation mark	PunctType=quot PunctSide=fin
PUNCT	``	``	opening quotation mark	PunctType=quot PunctSide=ini
PUNCT	HYPH	HYPH	punctuation mark, hyphen	PunctType=dash
PUNCT	LS	LS	list item marker	NumType=ord
PUNCT	NFP	NFP	superfluous punctuation	

PUNCT		LS	list item marker	NumType=ord
PUNCT		NFP	superfluous punctuation	
SYM	symbol	#	symbol, number sign	SymType=numbersign
SYM		$	symbol, currency	SymType=currency
SYM		SYM	symbol	
VERB	verb	BES	auxiliary "be"	
VERB		HVS	forms of "have"	
VERB		MD	verb, modal auxiliary	VerbType=mod
VERB		VB	verb, base form	VerbForm=inf
VERB		VBD	verb, past tense	VerbForm=fin Tense=past
VERB		VBG	verb, gerund or present participle	VerbForm=part Tense=pres Aspect=prog
VERB		VBN	verb, past participle	VerbForm=part Tense=past Aspect=perf
VERB		VBP	verb, non-3rd person singular present	VerbForm=fin Tense=pres
VERB		VBZ	verb, 3rd person singular present	VerbForm=fin Tense=pres Number=sing Person=3
X	other	ADD	email	
X		FW	foreign word	Foreign=yes
X		GW	additional word in multi-word expression	
X		XX	unknown	
SPACE	space	_SP	space	
		NIL	missing tag	

و هنا العديد من قيم كل منها :

pos

'adjective',
 'adposition',
 'adverb',
 'auxiliary',
 'coordinating conjunction',
'determiner',
'interjection',
'noun',
'numeral',
'other',
'particle',
'pronoun',
'proper noun',
'punctuation',

'space',
'subordinating conjunction',
'symbol',
'verb'}

dep
{None,
'adjectival complement',
'adjectival modifier',
'adverbial clause modifier',
'adverbial modifier',
'agent',
'appositional modifier',
'attribute',
'auxiliary',
'auxiliary (passive)',

'case marking',
'clausal complement',
'clausal modifier of noun (adjectival clause)',
'clausal subject',
'clausal subject (passive)',
'complement of preposition',
'compound',
'conjunct',
'coordinating conjunction',
'dative',
'determiner',
'direct object',
'expletive',
'interjection',
'marker',
'meta modifier',
'modifier of nominal',
'modifier of quantifier',

'negation modifier',
'nominal subject',
'nominal subject (passive)',
'noun phrase as adverbial modifier',
'numeric modifier',
'object of preposition',
'object predicate',
'open clausal complement',
'parataxis',
'particle',
'possession modifier',
'pre-correlative conjunction',
'prepositional modifier',
'punctuation',
'relative clause modifier',
'unclassified dependent'}

tag
{None,
 'adjective',
 'adjective, comparative',
 'adjective, superlative',
 'adverb',
 'adverb, comparative',
 'adverb, particle',
'adverb, superlative',
'cardinal number',
'closing quotation mark',
'conjunction, coordinating',
'conjunction, subordinating or preposition',
'determiner',
'email',
'existential there',
'foreign word',

'infinitival "to"',
'interjection',
'left round bracket',
'list item marker',
'noun, plural',
'noun, proper plural',
'noun, proper singular',
'noun, singular or mass',
'opening quotation mark',
'possessive ending',
'predeterminer',
'pronoun, personal',
'pronoun, possessive',
'punctuation mark, colon or ellipsis',
'punctuation mark, comma',
'punctuation mark, hyphen',
'punctuation mark, sentence closer',
'right round bracket',

'superfluous punctuation',
'symbol',
'symbol, currency',
'unknown',
'verb, 3rd person singular present',
'verb, base form',
'verb, gerund or present participle',
'verb, modal auxiliary',
'verb, non-3rd person singular present',
'verb, past participle',
'verb, past tense',
'wh-adverb',
'wh-determiner',
'wh-pronoun, personal'}

و هنا المزيد من التفاصيل

https://spacy.io/api/annotation#pos-tagging

-

ومن الممكن عمل فور تظهر التفاصيل كاملة هنا :

```
for token in doc1:
    print(f'{token.text:{10}} {token.pos_:{8}} {token.tag_:{6}} {spacy.explain(token.tag_)}')
```

و الشئ الهام ان مكتبة spacy لديها القدرة علي التمييز بين زمن الفعل , حتي لو كان التصريف متشابه

فهنا مثلا فعل مضارع :

```
doc = nlp(u'I read book now.')
r = doc[1]
print(f'{r.text:{10}} {r.pos_:{8}} {r.tag_:{6}} {spacy.explain(r.tag_)}')
```

بينما هنا ماضي

```
doc = nlp(u'I read a book on NLP.')
```

```
r = doc[1]

print(f'{r.text:{10}} {r.pos_:{8}} {r.tag_:{6}} {spacy.explain(r.tag_)}')
```

و يمكن معرفة نوع كل كلمة هكذا :

```
POS_counts = doc1.count_by(spacy.attrs.POS)
for k,v in sorted(POS_counts.items()):
    print(f'{k}. {doc1.vocab[k].text:{5}}: {v}')
```

و هنا التاجز

```
TAG_counts = doc1.count_by(spacy.attrs.TAG)
for k,v in sorted(TAG_counts.items()):
    print(f'{k}. {doc1.vocab[k].text:{4}}: {v}')
```

و الـ dep

```
DEP_counts = doc1.count_by(spacy.attrs.DEP)
```

```
for k,v in sorted(DEP_counts.items()):
    print(f'{k}. {doc1.vocab[k].text:{4}}: {v}')
```

-

كما يمكن ايضا استخدام nltk لنفس الهدف هكذا :

```
import nltk
from nltk.corpus import state_union
from nltk.tokenize import PunktSentenceTokenizer
```

```
text = 'Moses supposes his toeses are roses but moses supposes erroneously '

for w , m in nltk.pos_tag(nltk.word_tokenize(text)):
    print(f'word : ({w}), type : ({m}) , means :  ({spacy.explain(m)})')
```

مثال آخر

```
text = '''
Thomas Gradgrind, sir.  A man of realities.  A man of facts and calculations.  A man who proceeds upon the principle that two and two are four, and nothing over, and who is not to be talked into allowing for anything over.
```

Thomas Gradgrind, sir—peremptorily Thomas—Thomas Gradgrind. With a rule and a pair of scales, and the multiplication table always in his pocket, sir, ready to weigh and measure any parcel of human nature, and tell you exactly what it comes to. It is a mere question of figures, a case of simple arithmetic. You might hope to get some other nonsensical belief into the head of George Gradgrind, or Augustus Gradgrind, or John Gradgrind, or Joseph Gradgrind (all supposititious, non-existent persons), but into the head of Thomas Gradgrind—no, sir!

In such terms Mr. Gradgrind always mentally introduced himself, whether to his private circle of acquaintance, or to the public in general. In such terms, no doubt, substituting the words ‘boys and girls,’ for ‘sir,’ Thomas Gradgrind now presented Thomas Gradgrind to the little pitchers before him, who were to be filled so full of facts.
'''

نقوم اولا بعمل تقطيع للجمل

```
custom_sent_tokenizer = PunktSentenceTokenizer(text)
tokenized = custom_sent_tokenizer.tokenize(text)
tokenized[:10]
```

ثم عمل pos لكل كلمة في كل جملة

```
for i in tokenized[:5]:
    for w , m in nltk.pos_tag(nltk.word_tokenize(i)):
        print(f'word : ({w}), type : ({m}) , means :  ({spacy.explain(m)})')
```

```
    print('-----------------------------------------')
```

كما يمكن قراءة ملفات من هنا

```
import re
train_text = state_union.raw("2005-GWBush.txt")
sample_text = state_union.raw("2006-GWBush.txt")
```

و تقطيعها لجمل

```
custom_sent_tokenizer = PunktSentenceTokenizer(train_text)
tokenized = custom_sent_tokenizer.tokenize(sample_text)
tokenized[:5]
```

و تكرار الامر

```
for i in tokenized[:5]:
    for w , m in nltk.pos_tag(nltk.word_tokenize(i)):
        print(f'word : ({w}), type : ({m}) , means :  ({spacy.explain(m)})')
    print('-----------------------------------------')
```

لكن نتيجتها في اللغة العربية ضعيفة للغاية