**POS & NER tagging**

**POS-taggers:**

Knowledge of POS-tags can help us with a more in-depth text analysis. It is a pillar of natural language processing, and after the tokenizing text is usually the first piece of analysis which we carry out.

In the case of training a **spaCy model**, the **machine learning model** which we use to train the **POS-tagger** was **abstracted** to us. We only **used** the **update()** method to **train** our model, and don't know about the **nature** of the model, apart from the fact that it works well, and is a **neural network**.

*You’ll find an example of training a POS tagger in the textbook pages: 76,77,78*

The following code snippets illustrate some of the simple tasks we can do with POS-tags.

1. Transform certain words (example: verbs) into uppercase:

def make\_verb\_upper(text, pos):

    return text.upper() if pos == "VERB" else text

doc = nlp(u'Tom ran swiftly and walked slowly')

text = ''.join(make\_verb\_upper(w.text\_with\_ws, w.pos\_) for w in doc)

print(text)

1. Count the occurrences of certain tags:

import pandas as pd

harry\_potter = open("HP1.txt").read()

hp = nlp(harry\_potter)

hpSents = list(hp.sents)

hpSentenceLengths = [len(sent) for sent in hpSents]

[sent for sent in hpSents if len(sent) == max(hpSentenceLengths)]

hpPOS = pd.Series(hp.count\_by(spacy.attrs.POS))/len(hp)

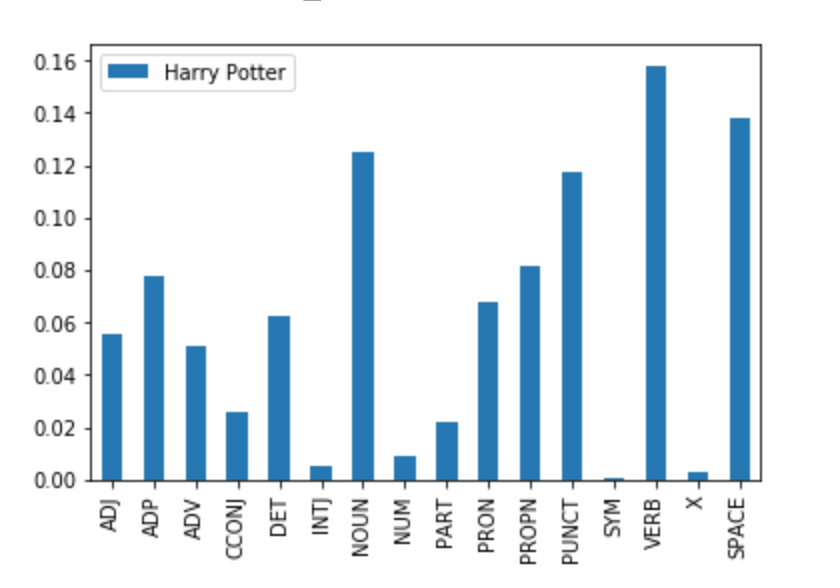
tagDict = {w.pos: w.pos\_ for w in hp}

hpPOS = pd.Series(hp.count\_by(spacy.attrs.POS))/len(hp)

df = pd.DataFrame([hpPOS], index=['Harry Potter'])

df.columns = [tagDict[column] for column in df.columns]

df.T.plot(kind='bar')



1. If we want to find the most commonly used pronouns, we only need two lines

hpAdjs = [w for w in hp if w.pos\_ == 'PRON']

Counter([w.string.strip() for w in hpAdjs]).most\_common(10)

**NER-taggers:**

NER stands for **Named Entity Recognition**, and along with part of speech tagging, it is one of the pillars of natural language processing.

A named entity is a real-world object with a proper name – examples are Palestine, Donald Trump, and TikTok.

In these examples, Palestine is a **country** and would be identified as a GPE (Geopolitical Entity), Donald Trump as PER (a **person**), and TikTok is a **company**, so identified as an ORG (Organization).

How many different categories of named entities exist? We can choose to be vague with our entities, only recognizing a few, or have a really fine-grained set of categories. Most modern NER-taggers, similar to POS-taggers, are statistically trained models where the number of classes is equal to the number we want them to be, and depending on the problem.

There are a few categories: person ( PER ), location ( LOC ), organization ( ORG ), and other miscellaneous entities ( MISC ). spaCy features 18 different categories for its named entity classification.

**Why should we now be interested in NER-tagging?** As usual, simply identifying named entities in text is not often the end result of our task, but it ends up being an important building block for further tasks. Entity linking is a task where we use entity recognition and then attempt to derive relationships between them. Consider this sentence:

Rome is the capital of Italy.

Any NER-tagger would recognize Rome as a place (GPE), as well as Italy. To be able to draw the conclusion that Rome is a city, which is linked to the country Italy, are the kind of tasks that we call as **Named Entity Disambiguation (NED).**

This is also of great value in **biomedical research**, where scientists attempt to identify genes and gene products. It can be used by the **businesses** to help identify which organizations are most important by analyzing and identifying links between other organizations and revenue. Both of these examples are domain-specific though; do not expect a tagger trained on medical journal data to perform well on financial documents!

The **most popular** **usage** of NER-tagging in science still remains in the field of **medicine and biology**, which is also evident by the existence of competitions just devoted to extracting entities from medical documents.

**NER-tagging in python:**

For example, consider the following sentence:

The little brown dog barked at the black cat.

We can identify the two noun phrases quite easily: the little brown dog and the black cat. These **chunks** can come in handy when we're doing NER-tagging.

*So why exactly is it relevant? Donald Trump would be tagged as a person; not just Donald, or Trump, but the entire phrase. This knowledge of a group of words as a noun phrase can help make decisions when we are tagging.*

spaCy uses an online tagging system is called BILOU.

import spacy

nlp = spacy.load('en')

sent\_0 = nlp(u'Donald Trump visited at the government headquarters in France today.')

sent\_1 = nlp(u'Emmanuel Jean-Michel Frédéric Macron is a French politician serving as President of France and ex officio Co-Prince of Andorra since 14 May 2017.')

sent\_2 = nlp(u"He studied philosophy at Paris Nanterre University, completed a Master's of Public Affairs at Sciences Po, and graduated from the École nationale d'administration (ÉNA) in 2004.")

sent\_3 = nlp(u'He worked at the Inspectorate General of Finances, and later became an investment banker at Rothschild & Cie Banque.')

#printing tokens and their ner tag

for token in sent\_0:

    print(token.text, token.ent\_type\_)

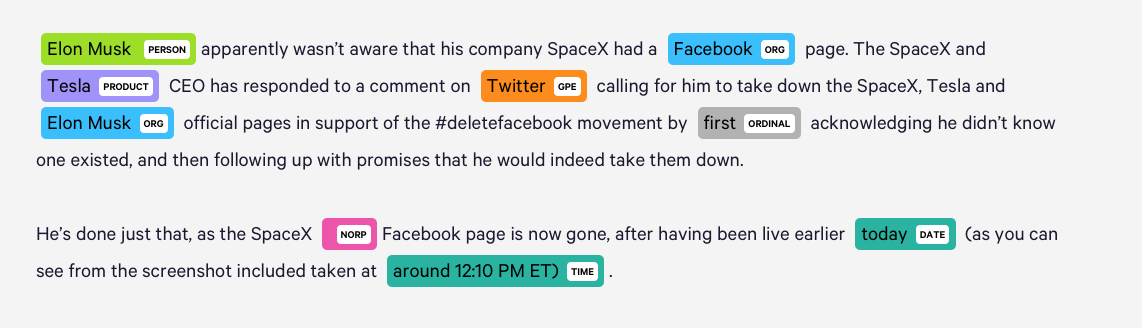
#words that were not identified as named entities, an empty string is returned

#printing entities and their ner tag

for ent in sent\_0.ents:

    print(ent.text, ent.label\_)

*An example of training a NER tagger is given in the textbook pages 96,97,98,99,100,101.*



**Homework :**

Using the textbook examples of training a POS and NER tagger, picking a corpus of your choice:

1. train a pos tagger to recognize a new tag “NNS” associated to plural nouns,

"I love cats"

1. and a ner tagger to recognize a new entity label called “TECH” associated to Python

“I’m learning python”