COMP6713 - Natural Language Processing – 24T1

Tutorial – Week 2 with solutions

Installing NLTK: https://www.nltk.org/install.html
Installing SpaCy: https://spacy.io/usage
Installing Transformers: https://huggingface.co/docs/transformers/en/installation

Q. 1. Compare and analyse the output of NLTK and spaCy POS taggers for the following sentences:
1. The cat is on the mat

2. "Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo" is a grammatically correct English sentence. It is often used as a stress test for POS taggers. [1]

3. "The old man the boat"
What errors do the POS taggers make?

NLTK:
```
import nltk
from nltk import word_tokenize
text = word_tokenize("The cat is on the mat.")
result = nltk.pos_tag(text)
result
```

SpaCy:
```
import spacy
text = "The cat is on the mat."
nlp = spacy.load("en_core_web_sm")
doc = nlp(text)
for token in doc:
   print(token.text, token.pos_)
```

Observations: The two libraries use different tag sets – so the outputs are different. Both POS taggers get the sentences (b) and (c) wrong. They are truly the edge cases of POS taggers.

Q. 2. Masked language modelling (MLM) is a form of self-supervision. A dataset of unlabeled sentences can be converted to a self-supervised MLM task as follows: Randomly mask a word in a sentence. The input to the language model is the sentence with the word masked, the output is the word at the masked position in the original sentence.
The sentences in the dataset are three quotes by Oscar Wilde:

(i) You can never be overdressed or overeducated.
(ii) Always forgive your enemies; nothing annoys them so much.
(iii) I am not young enough to know everything.

(A) Draw the black-box view to create a self-supervised MLM task as described above.
(B) Use the HuggingFace pipeline 'fill-mask' to obtain the top 5 words for each of the sentences above.
(C) Calculate precision@5. Recall from the lecture that precision is the number of correctly retrieved words. "@5" indicates that any of the top 5 words need to be correct.

(a)

You can never be _____ or overeducated. ----> overdressed

Always forgive your _____; nothing annoys them so much. ---> enemies

I am not _____ enough to know everything. ---> young

(Note that any other word can be masked out. So, multiple alternatives are possible.)

(b) from transformers import pipeline

masklm = pipeline("fill-mask", model='roberta-base', topk=5)

words = masklm("You can never be _____ or overeducated.")

(c) Call masklm to get the predicted word for the masked position for the three sentences. Precision@5 is the proportion of times the missing word is actually retrieved. Denominator will be 3 since there are three sentences.

Extra:

Q. 3. Use the spaCy matcher code provided in the class and extend it to also return the WordNet lemmas of words that are extracted as entities.

Q. 4. Work through the NLTK tutorial at: https://www.nltk.org/book/ch06.html