Assignment 1 - Report

Thomas Reolon - 221247

5 FUNCTIONS WITH SPACY (probably reading the notebook is easier)

1. extract a path of dependency relations from the ROOT to a token T

function: get_path_from_root
input: sentence:str
output: dict[Token --> list[depencency]

Each token has an attribute head which is a pointer to its parent, we can rebuild the path from the **ROOT** token to **T** as the reverse path of starting from **T** and following the head pointer to the **ROOT**. We know we reached the root because it has no parent (root.head is a self-loop).

2. extract subtree of a dependents given a token

function: get_subtree
input: sentence:str
output: dict[Token --> list[token_str]]

SpaCy offers a built-in function (token.subtree) that returns a generator of the subtree

3. check if a given list of tokens (should be a span) forms a subtree

function: is_subtree
input: sentence:str, list_of_tokens:List[str]
output: bool

We get a list of all the possible subtrees using the previous function, then, if list_of_token is a subtree there will be a list containing the same tokens inside the list of all the possible subtrees

4. identify head of a span, given its tokens

function: get_head_of_span
input: span [list or Span or ...]
output: spacy.tokens.Token

Span.root returns the head of the span if the span is a subtree. This function implement some casting to handle different types of inputs.

5. extract sentence subject, direct object and indirect object spans

function: extract_sub_ind_dir
input: sentence:str
output: dict[element --> list[token_str]]

Token.dep_ contains the relation of a dependency. We can find subject direct_object and indirect_object checking what value is contained in dep_ attribute for each Token.

DEPENDENCY PARSER WITH NLTK

We can modify nltk default TransitionParser extending its classes.

Extend `nltk.parse.transitionparser.Configuration` overriding the method extract_features to extract our features

Lower tokens text

```
if self._check_informative(token["word"], True):
    result.append("BUF_0_FORM_" + token["word"].lower())
```

Add info on dependency arc as feature

Removed lemma from the used features (it seemed too much similar to word attribute, removing it greatly improved results).

I also tried other changes, eg. incrementing the number of buffer processed tokens from 4 to 5, but it did not improve

Extend `nltk.parse.transitionparser.TransitionParser` overriding the method train to specify our model and _create_training_samples to specify to use our Configuration class

```
model = svm.SVC(
    kernel="rbf",
    coef0=0,
    gamma=0.08, # ---> low = higher variance (easier similarity)
    C=4,        # ---> high = high penalization for errors
    verbose=verbose,
    probability=True,
)
```