You can read the full article at its original source on the Towards Data Science website here.

Coreference Resolution in Python

Integrate Neural Network-Based Coreference Resolution into your NLP Pipeline using NeuralCoref

In human language, endophoric awareness plays a key part in comprehension (decoding) skills, writing (encoding) skills, and general linguistic awareness. Endophora consists of anaphoric, cataphoric, and self-references within a text.

Anaphoric references occur when a word refers back to other ideas in the text for its meaning.

```
David went to the concert. He said it was an amazing experience.

He refers to David.
It refers to the concert.
```

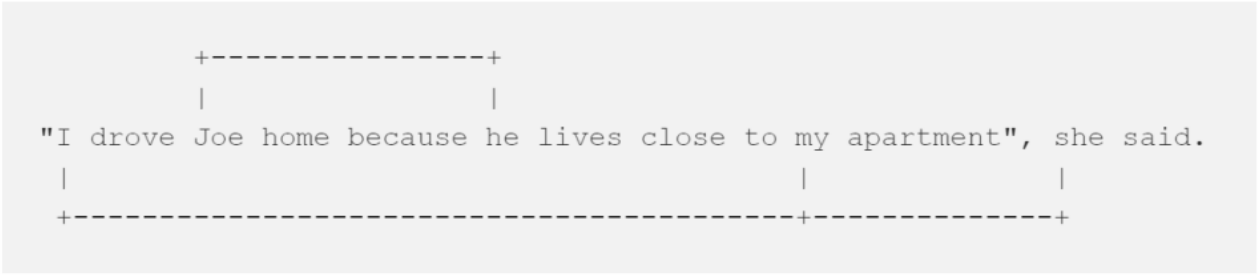
Cataphoric references occur when a word refers to ideas later in the text.

```
Every time I visit her, my grandma bakes me cookies.

Her refers to my grandma.
```

Coreference resolution is the NLP (Natural Language Processing) equivalent of endophoric awareness used in information retrieval systems, conversational agents, and virtual assistants like Amazon's Alexa. It is the task of clustering mentions in text that refer to the same underlying entities.

For example:



"I", *"my"*, and *"she"* belong to the same cluster and *"Joe"* and *"he"* belong to the same cluster.

Algorithms which resolve coreferences commonly look for the nearest preceding mention that is compatible with the referring expression. Instead of using rule-based dependency parse trees, neural networks can also be trained which take into account word embeddings and distance between mentions as features.

NeuralCoref is an open source python packgage integrated in SpaCy's NLP pipeline. You can install NeuralCoref with pip:

```
pip install neuralcoref
```

or from sources with dependencies in a virtual environment:

```
venv .env
source .env/bin/activate
git clone https://github.com/huggingface/neuralcoref.git
cd neuralcoref
pip install -r requirements.txt
pip install -e .
```

SpaCy and NeuralCoref can be used to create production-ready NLP applications with little fine-tuning. For example, let's parse through the historical United States v. Nixon case to retrieve facts referencing the former U.S. President Richard Nixon:

```
import urllib.request
from bs4 import BeautifulSoup
import spacy
import neuralcoref
nlp = spacy.load('en_core_web_lg')
neuralcoref.add_to_pipe(nlp)
```

```
html =
urllib.request.urlopen('https://www.law.cornell.edu/supremecourt/text/418/683
').read()
soup = BeautifulSoup(html, 'html.parser')
text = ''.join([t for t in soup.find_all(text=True) if t.parent.name == 'p'
and len(t) >= 25])
doc = nlp(text)
resolved_text = doc._.coref_resolved
sentences = [sent.string.strip() for sent in nlp(resolved_text).sents]
output = [sent for sent in sentences if 'president' in
          (' '.join([token.lemma_.lower() for token in nlp(sent)]))]
print('Fact count:', len(output))
for fact in range(len(output)):
    print(str(fact+1)+'.', output[fact])
```

Output:

Fact count:108

1. *Following indictment alleging violation of federal statutes by certain staff members of the White House and political supporters of the President, the Special Prosecutor filed a motion under Fed.*
2. *Proc. 17(c) for a subpoena for the production before trial of certain tapes and documents relating to precisely identified conversations and meetings between the President and others.*
3. *the President, claiming executive privilege, filed a motion to quash the subpoena.*

The script scrapes the webpage with Urllib and parses HTML using Beautiful Soup. We load the text into a SpaCy model of our choice; you can download pre-trained SpaCy models from the terminal as shown below:

```
python -m spacy download en_core_web_lg
```

The SpaCy pipeline assigns word vectors, context-specific token vectors, part-of-speech tags, dependency parsing, and named entities. by extending the SpaCy's pipeline of annotations you can resolve coreferences.

You can retrieve a list of all the clusters of corefering mentions using the `doc._.coref_clusters`attribute and replace corefering mentions with the main mentions in each cluster by using the `doc._.coref_resolved` attribute.

The SpaCy pipeline assigns word vectors, context-specific token vectors, part-of-speech tags, dependency parsing, and named entities. by extending the SpaCy's pipeline of annotations you can resolve coreferences.

You can retrieve a list of all the clusters of corefering mentions using the `doc._.coref_clusters`attribute and replace corefering mentions with the main mentions in each cluster by using the`doc._.coref_resolved`attribute.

SpaCy has a built-in unsupervised sentence tokenizer to split the text into a list of sentences. Use lowercased lemmatized sentences for approximate string searching to the topic of your interest (e.g. President).

You can read the published article on Towards Data Science here.

Share this:

- Click to share on Twitter (Opens in new window)