AASD 4004
Machine Learning - II

Applied AI Solutions Developer Program

Module 04
Information Extraction

Vejey Gandyer

Agenda

- Information Extraction
- IE Pipeline
- Key Phrase Extraction
- Named Entity Recognition
- Named Entity Disambiguation & Linking
- Relation Extraction
- Event Extraction ?????

Information Extraction

What is it?

Information Extraction

Refers to the NLP task of extracting relevant information from text documents

Information - Key events, entities, people, relationships, ...

Types

Key Phrase Extraction

Named Entity Recognition

Named Entity Disambiguation & Linking *

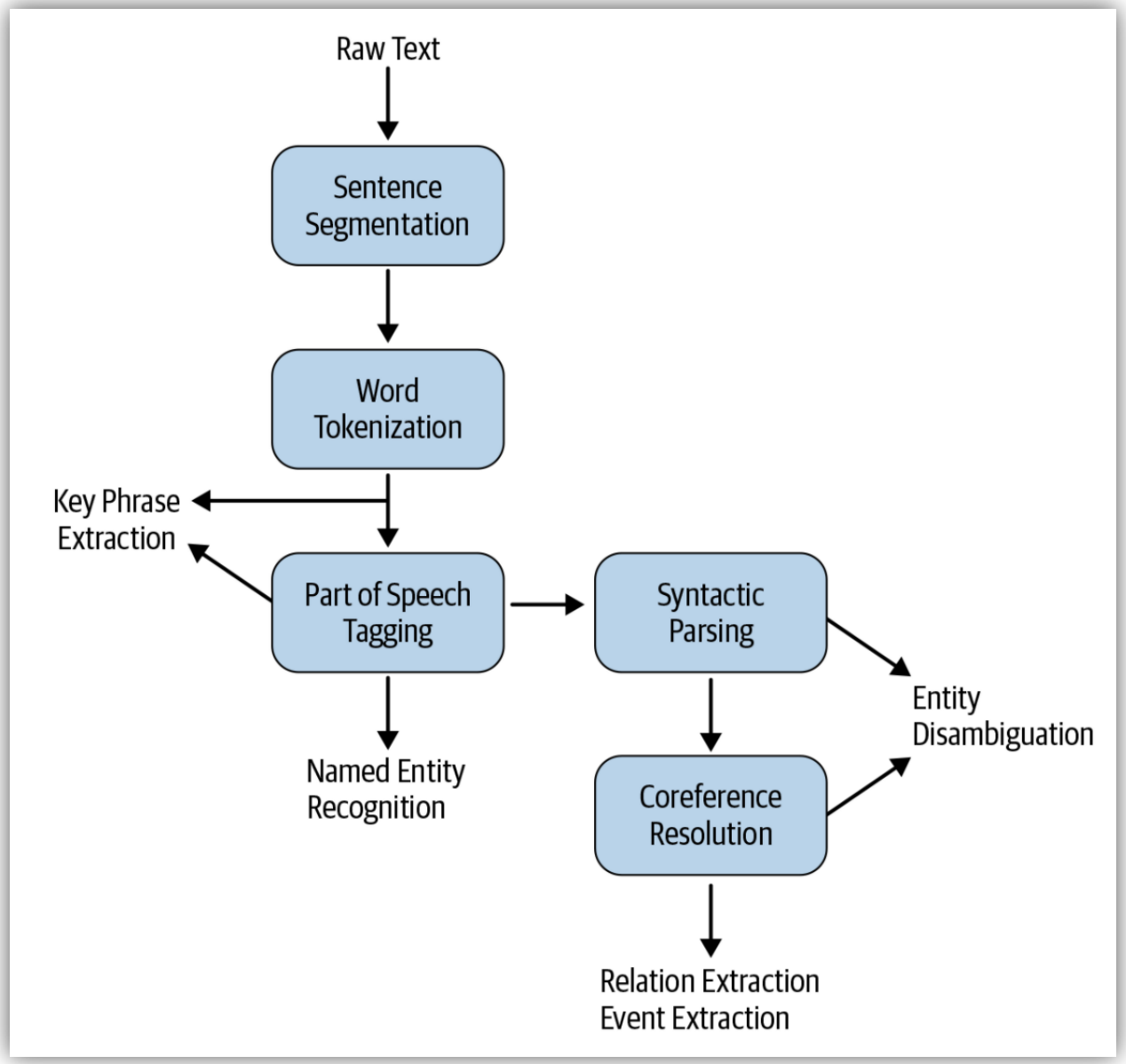
Relation Extraction *

Event Extraction *

* - Will be seen in DL-I

IE Pipeline

IE Pipeline


Raw Text
Sentence Segmentation
Word Tokenization
Key Phrase Extraction
Part of Speech Tagging
Named Entity Recognition
Syntactic Parsing
Entity Disambiguation
Coreference Resolution
Relation Extraction
Event Extraction

Key Phrase Extraction

Key Phrase Extraction

Task of extracting important words and phrases that capture the gist of the text from a given text document

Applications

- Information retrieval
- Automatic document tagging
- Recommendation system
- Text summarization

Approaches

- Supervised
  - Needs labeled data
  - Time-consuming labeling process
- Unsupervised
  - No labels needed

Key Phrase Extraction - Unsupervised

Based on Graph approach

Node - Keywords

Edge - Weights (importance)

Top-N keywords are those nodes with well connected with other nodes

Key Phrase Extraction - textacy

```
import spacy
import textacy.ke
from textacy import *
```

```
en = textacy.load_spacy_lang("en_core_web_sm")
path = 'PATH TO DATASET'
mytext = open(path+'./Data/nlphistory.txt').read()
doc = textacy.make_spacy_doc(mytext, lang=en)
```

```
#Print the keywords using TextRank algorithm, as implemented in Textacy.
print("Textrank output: ", [kps for kps, weights in textacy.ke.textrank(doc, normalize="lemma", topn=5)])
#Print the key words and phrases, using SGRank algorithm, as implemented in Textacy
print("SGRank output: ", [kps for kps, weights in textacy.ke.sgrank(doc, topn=5)])
```

```
#To address the issue of overlapping key phrases, textacy has a function: aggregage_term_variants.
#Choosing one of the grouped terms per item will give us a list of non-overlapping key phrases!
terms = set([term for term,weight in textacy.ke.sgrank(doc)])
print(textacy.ke.utils.aggregate_term_variants(terms))
```

```
#A way to look at key phrases is just consider all noun chunks as potential ones.
#However, keep in mind this will result in a lot of phrases, and no way to rank them!
print([chunk for chunk in textacy.extract.noun_chunks(doc)])
```

Key Phrase Extraction - textacy

- The process of extracting potential n-grams and building the graph with them is sensitive to document length, which could be an issue in a production scenario. One approach to dealing with it is to not use the full text, but instead try using the first M% and the last N% of the text, since we would expect that the introductory and concluding parts of the text should cover the main summary of the text.
- Since each keyphrase is independently ranked, we sometimes end up seeing overlapping keyphrases (e.g., "buy back stock" and "buy back"). One solution for this could be to use some similarity measure (e.g., cosine similarity) between the top-ranked keyphrases and choose the ones that are most dissimilar to one another. textacy already implements a function to address this issue, as shown in the notebook.
- Seeing counterproductive patterns (e.g., a keyphrase that starts with a preposition when you don't want that) is another common problem. This is relatively straightforward to handle by tweaking the implementation code for the algorithm and explicitly encoding information about such unwanted word patterns.
- Improper text extraction can affect the rest of the KPE process, especially when dealing with formats such as PDF or scanned images. This is primarily because KPE is sensitive to sentence structure in the document. Hence, it's always a good idea to add some post-processing to the extracted key phrases list to create a final, meaningful list without noise.

Named Entity Recognition

NER

Named Entity Recognition - Motivation

Where was Albert Einstein born?

Answer: Ulm, Germany

Albert Einstein --> Entity (PERSON)

NER refers to the task of identifying entities in a document

Named Entity Recognition using spaCy

SpaCy - Built-in NER capability

<u>Issues</u>

- No custom entities can be detected apart from PERSON, PLACE, ORG and so on
- Unusal Sentence Splitting (sometimes)

Custom Named Entity Recognition System

Import necessary libraries

Load the dataset - CONLL

Extract features

Train CRF model

Evaluate CRF model

Named Entity Disambiguation & Linking

NED & NEL

Named Entity Disambiguation (NED)

Lincoln drives a Lincoln Aviator and lives on Lincoln Way

All three mentions of "Lincoln" refers to different entities

NED refers to tha task of assigning a unique identity to entities mentioned in the text. It maps to corresponding Knowledge Bases (Google Knowledge Graph, Wikipedia Knowledge Graph, …)

Named Entity Linking (NEL)

NEL = NER + NED

Azure Text Analytics API performs NEL

Input: Text

Output: Entities detected with its corresponding Wikipedia links

```
Entities in this document:
San Francisco    Location
https://en.wikipedia.org/wiki/San_Francisco
Facebook         Organization
https://en.wikipedia.org/wiki/Facebook
Alex Jones       Person
https://en.wikipedia.org/wiki/Alex_Jones
InfoWars         Organization
https://en.wikipedia.org/wiki/InfoWars
Louis Farrakhan          Person
https://en.wikipedia.org/wiki/Louis_Farrakhan
Silicon Valley   Location
https://en.wikipedia.org/wiki/Silicon_Valley
Instagram        Organization
https://en.wikipedia.org/wiki/Instagram
us       Location
```

Relation Extraction

RE

Relation Extraction

RE refers to the task of extracting entities and relationships between entities from text documents

Set of triples (Subject, verb, Object)

Include: Self- Knowledge Creation notebooks

Satya Narayana Nadella is an Indian-American business executive. He currently serves as the Chief Executive Officer (CEO) of Microsoft, succeeding Steve Ballmer in 2014. Before becoming chief executive, he was Executive Vice President of Microsoft's Cloud and Enterprise Group, responsible for building and running the company's computing platforms.

Relation Extraction using Watson API

Watson API

```
employedBy
[{'type': 'Person', 'text': 'Nadella'}]
[{'type': 'Organization', 'text': 'Hyderabad Public School', 'disambiguation': {'subtype': ['Commercial']}}]
awardedTo
[{'type': 'Degree', 'text': 'bachelor'}]
[{'type': 'Person', 'text': 'Nadella'}]
educatedAt
[{'type': 'Person', 'text': 'Nadella'}]
[{'type': 'Organization', 'text': 'Manipal Institute of Technology', 'disambiguation': {'subtype': ['Educati
onal']}}]
educatedAt
[{'type': 'Person', 'text': 'Nadella'}]
[{'type': 'Organization', 'text': 'Mangalore University', 'disambiguation': {'subtype': ['Educational']}}]
awardedBy
[{'type': 'Degree', 'text': 'bachelor'}]
[{'type': 'Organization', 'text': 'Manipal Institute of Technology', 'disambiguation': {'subtype': ['Educati
onal']}}]
basedIn
[{'type': 'Organization', 'text': 'Mangalore University', 'disambiguation': {'subtype': ['Educational']}}]
[{'type': 'GeopoliticalEntity', 'text': 'Karnataka'}]
```

Exercise: Building an end-to-end Information Extraction System

KPE, NER

Building an IE System

Create an Information Extraction System that has the following features:

- KPE
- NER
- NEL*
- RE*

* DL-I

Further Reading

Practical Natural Language Processing

Soumya Vajjala, Anuj Gupta, Harshit Surana, Bodhisattwa Majumder