NLP tasks for Data Science

CMPT 733
Steven Bergner
Slides in part by: Suraj Swaroop (Summer coop, 2020)

What is NLP?

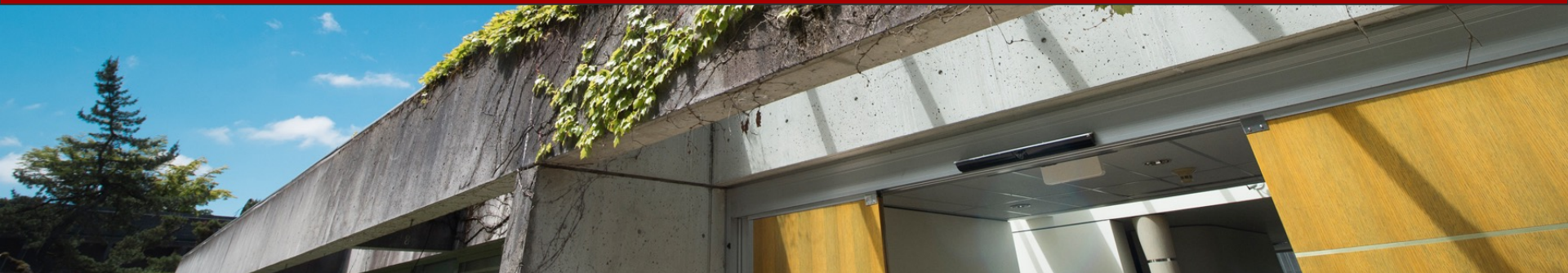
Natural Language

how humans communicate with each other via **speech and text**

Processing

- branch of AI to read, decipher, and make sense of human language
- Applications: information extraction, translation, personal assistants, word processors, spam detection, …

Techniques for NLP

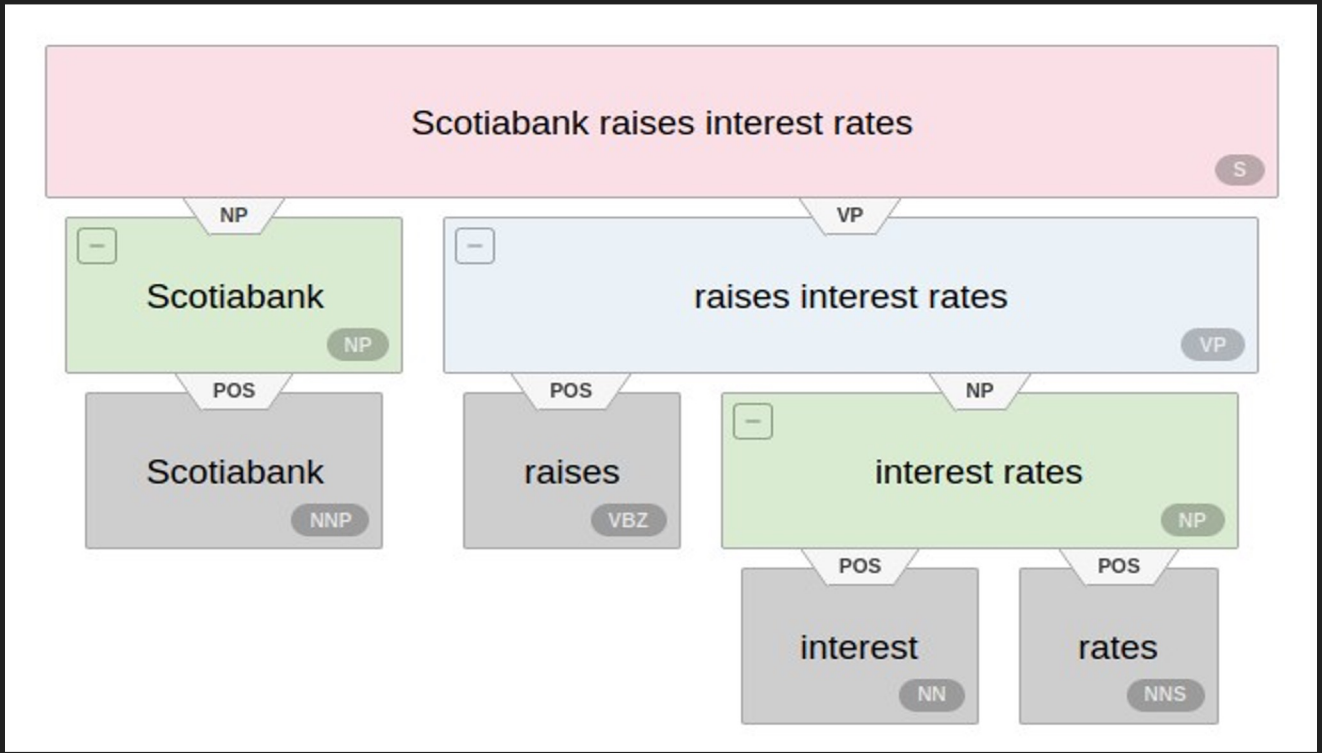


Text Parsing

- Analyzing sentence structure and representing it according to syntactic formalism

- Two views of syntactic structure
  - Constituency
  - Dependency

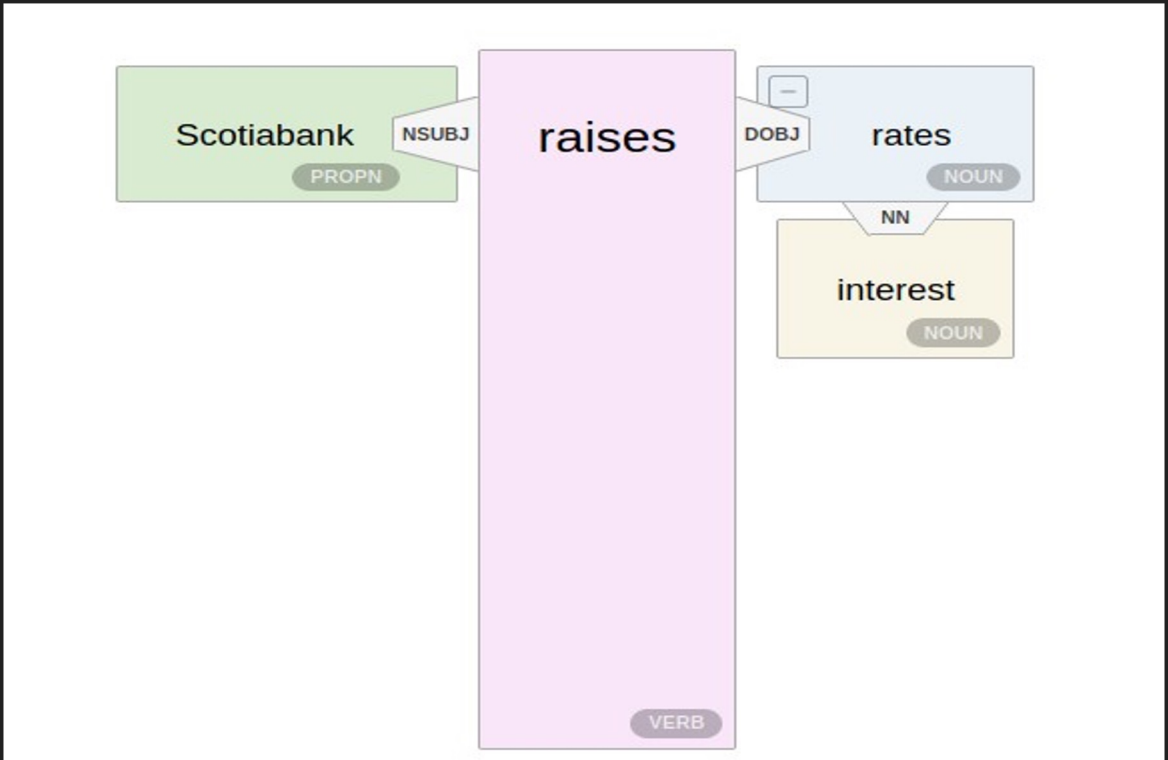
Constituency Structure Example



Constituency Parsing Implementation

```
from constituent_treelib import ConstituentTree, BracketedTree, Language, Structure

# Define the language for the sentence as well as for the spaCy and benepar models
language = Language.English

# Define which specific SpaCy model should be used (default is Medium)
spacy_model_size = ConstituentTree.SpacyModelSize.Medium

# Create the pipeline (note, the required models will be downloaded and installed automatically)
nlp = ConstituentTree.create_pipeline(language, spacy_model_size)

without_token_leaves = ConstituentTree(sentence, nlp, Structure.WithoutTokenLeaves)
```

Dependency Structure Example



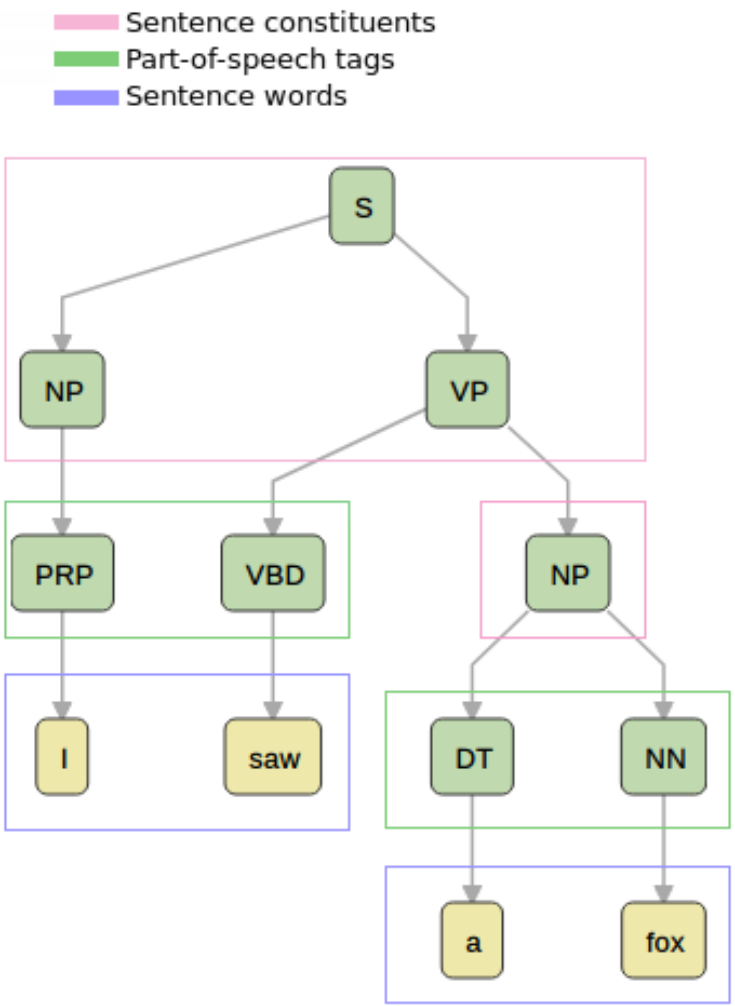
Dependency vs Constituency Tree

Dependency parsing

- only relationships between words and their constitutes

Constituency parsing

- entire sentence structure and relationships between phrases

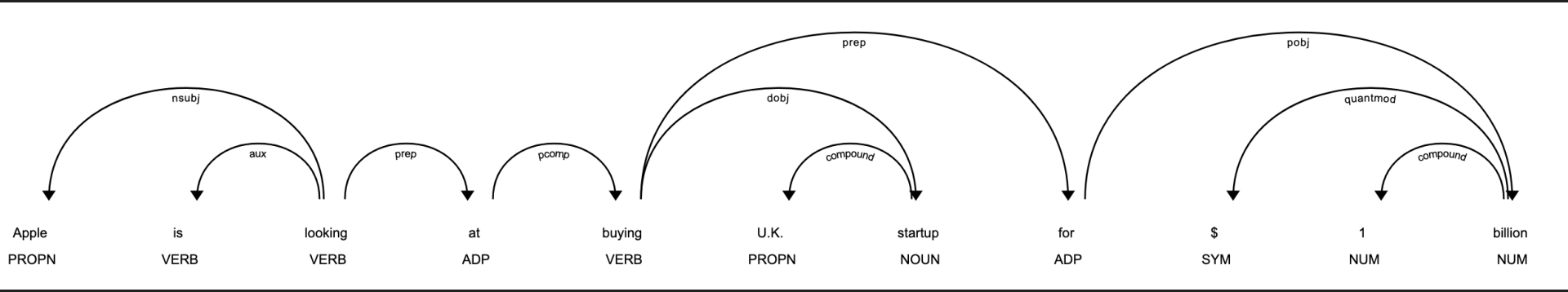


Phrase examples

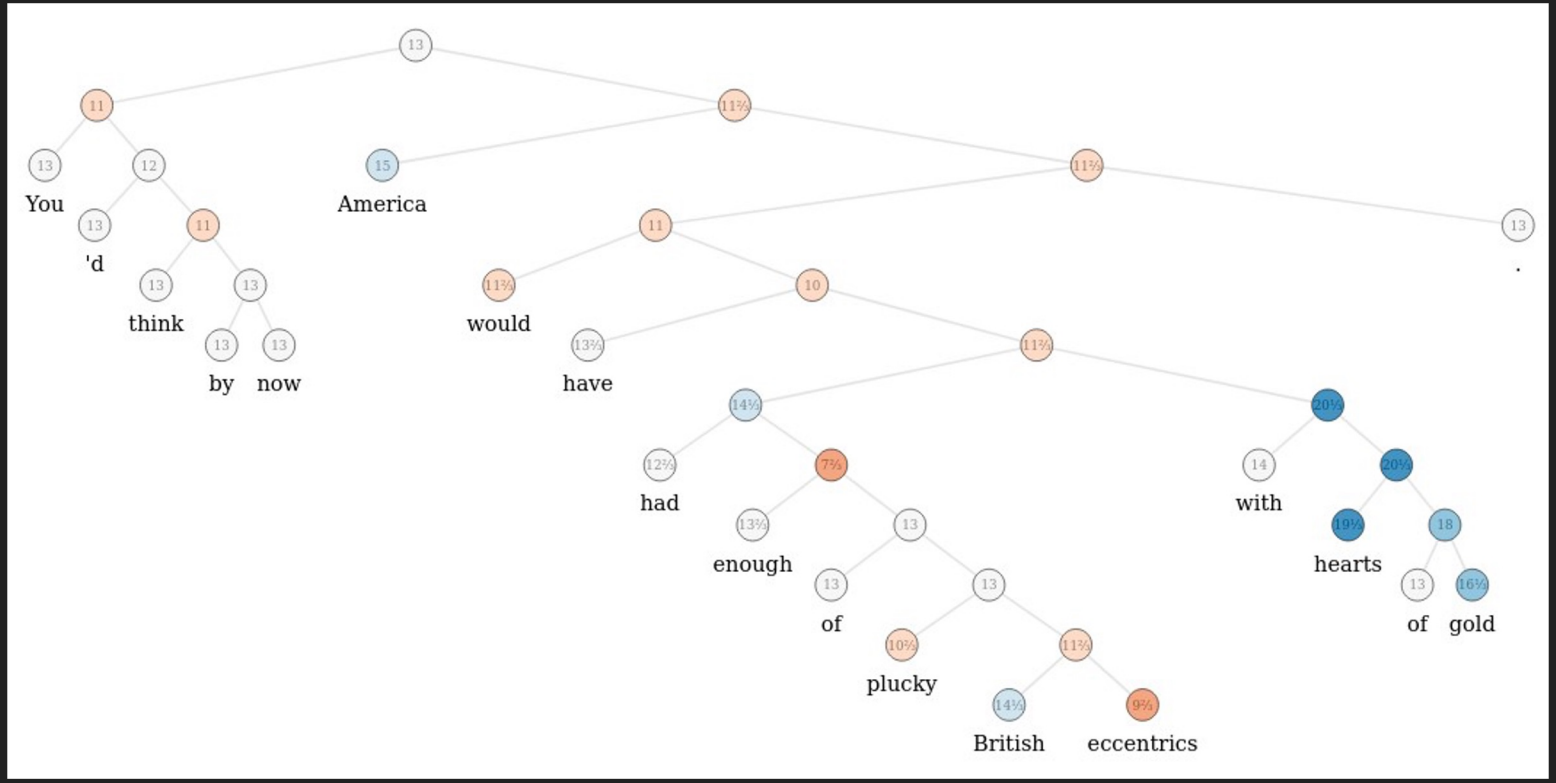
label	long name	example (represented by terminal string)
NP	noun phrase	their public lectures
VP	verb phrase	built the pyramid
PP	preposi-tional phrase	in the five chambers
S	sentence	Khufu built the pyramid
SBAR	sbar	that Khufu built the pyramid

Dependency tree

“Apple is looking at buying U.K. startup for $ 1 billion”



Tree Example [Stanford Sentiment Treebank]



Information Extraction

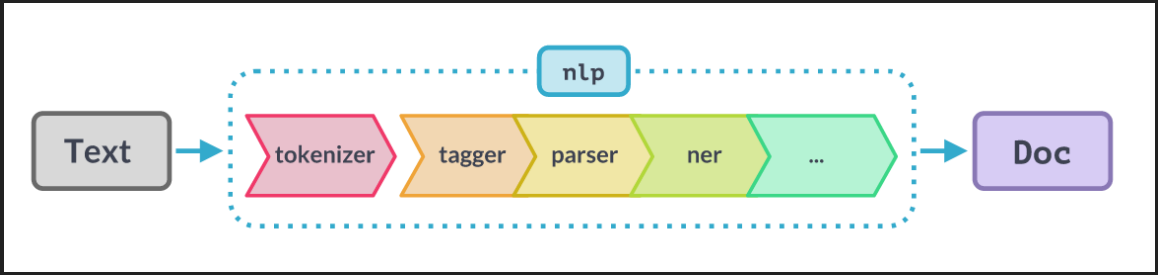
- Automatic extraction of structured and unstructured information

- Various modules
  - POS Tagging
  - Entity Recognition
  - Relation extraction
  - Sentiment Analysis

Named Entity Recognition

- Classify named entities into categories
- NER Techniques
  - Lexicon approach
  - Rule-based systems
  - ML based systems
  - Hybrid approach

NER Implementation



- **Text:** The original word text.
- **Lemma:** The base form of the word.
- **POS:** The simple UPOS part-of-speech tag.
- **Tag:** The detailed part-of-speech tag.
- **Dep:** Syntactic dependency, i.e. the relation between tokens.
- **Shape:** The word shape – capitalization, punctuation, digits.
- **is alpha:** Is the token an alpha character?
- **is stop:** Is the token part of a stop list, i.e. the most common words of the language?

```
nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying U.K. startup for $1 billion")

pd.DataFrame((token.text, token.lemma_, token.pos_, token.tag_, token.dep_,
              token.shape_, token.is_alpha, token.is_stop) for token in doc),
              columns="Text Lemma POS Tag Dep Shape alpha stop".split())
```

Sentiment Analysis

- Determine if an opinion is positive, negative or neutral

- Techniques for Sentiment Analysis
  - Lexical Methods
  - Machine Learning methods

Sentiment Analysis Implementation

```
[8] import nltk
    nltk.download('vader_lexicon')
    from nltk.sentiment.vader import SentimentIntensityAnalyzer
    sid = SentimentIntensityAnalyzer()
    sid.polarity_scores("I am happy today")
```

```
{'compound': 0.5719, 'neg': 0.0, 'neu': 0.351, 'pos': 0.649}
```

Part of speech Tagging

- Tags each word with its corresponding part of speech
- Techniques of POS
  - Lexical Based Methods
  - Rule Based Method
  - Probabilistic Method
  - Deep learning models

POS Tagging Implementation

(use spacy instead)

```
import nltk
from nltk import word_tokenize
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
text = word_tokenize("He would not accept anything of value from those he was writing about")
nltk.pos_tag(text)
```

```
[('He', 'PRP'),
 ('would', 'MD'),
 ('not', 'RB'),
 ('accept', 'VB'),
 ('anything', 'NN'),
 ('of', 'IN'),
 ('value', 'NN'),
 ('from', 'IN'),
 ('those', 'DT'),
 ('he', 'PRP'),
 ('was', 'VBD'),
 ('writing', 'VBG'),
 ('about', 'IN')]
```

Semantic Role Labeling (SRL)

- Assigning labels to words or phrases in a sentence to indicate it's semantic role

- How it works:
  - Predicate identification
  - Predicate disambiguation
  - Argument identification
  - Argument classification

For Example

“He wouldn’t accept anything of value from those he was writing about”

The annotations of semantic roles for this sentence:

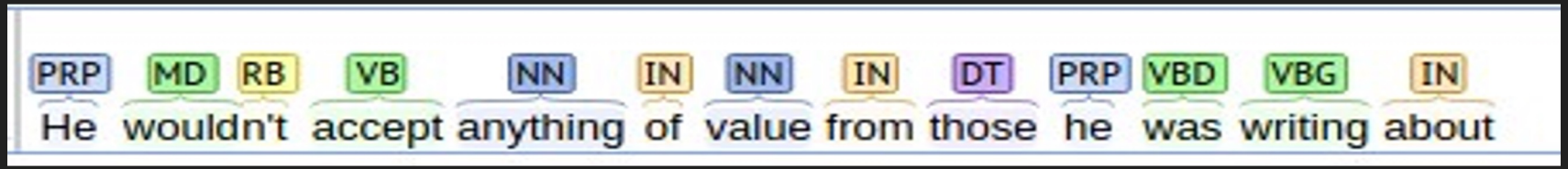
[$_{A0}$ He] [$_{AM\text{-}MOD}$ would] [$_{AM\text{-}NEG}$ n't] [$_{V}$ accept] [$_{A1}$ anything of value] from [$_{A2}$ those he was writing about] .

V: verb; A0: acceptor; A1: thing accepted; A2: accepted-from; A3: attribute;
AM-MOD: modal; AM-NEG: negation

Difference between POS and SRL

Sentence: “He wouldn’t accept anything of value from those he was writing about”

The annotations of POS Tagging:



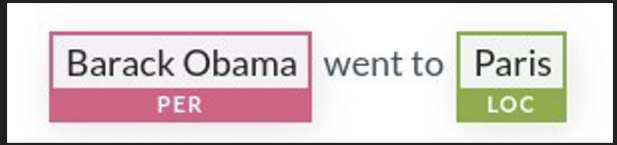
The annotations of semantic roles for this sentence:

[$_{A0}$ He] [$_{AM-MOD}$ would] [$_{AM-NEG}$ n't] [$_{V}$ accept] [$_{A1}$ anything of value] from [$_{A2}$ those he was writing about] .

NER and SRL

Sentence: “Barack Obama went to Paris “

The annotations of Entity Recognition Tagging:



The annotations of semantic roles for this sentence:

[ARG0: Barack Obama] [V: went] [ARG4: to Paris]

Combining ER and SRL

SA and NER

- Document-level sentiment analysis
  - Documents may have multiple topics
  - Not enough granularity

- Entity sentiment analysis identifies sentiment of each word
  - know how specific people, organizations, or things are being mentioned

Applications of SRL

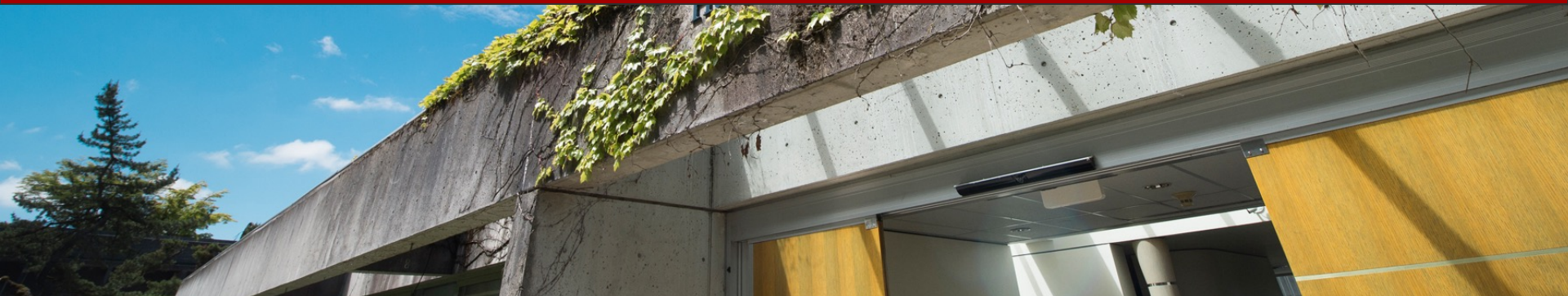
- Question Answering system
- Summarization
- Information Extraction

Tools

Tools for NLP

- **NLTK** - legacy baseline, dictionary and rule based methods
- **Spacy**
  - Supports several different languages
- **Huggingface transformers** [github/demos]
  - Many state-of-the-art pre-trained models
- **AllenNLP** - platform for solving natural language processing tasks in PyTorch
- **Torchtext** – text processing support for PyTorch

Thank You



Innovation Prize

Judged by Big Data External Industrial Advisory Panel

- 2 BD Project Winners, $2500 each
- 1 VC Project Winner, $2500
- 1 CY Project Winner, $2500