CIS 192: Python Programming

Data Science III:
Natural Language Processing

Arun Kirubarajan
kiruba@cis.upenn.edu
@kirubarajan
kirubarajan.com
Motivation: chatbots

a “holy grail” of computer science est. 1950
it’s a necessary condition (i.e. AI implies understanding natural language)
Thought Experiment:
How can we build a Python chatbot?
What is natural language?

How do humans learn natural language?
How do computers learn natural language?
Mathematically, teaching computers to understand natural language is more difficult than the Traveling Salesman Problem.

NLP has no known “solutions”, regardless of efficiency. So, we must use heuristics for natural language problems.

results > proofs
Philosophy Time:
Is understanding language sufficient or necessary for artificial intelligence?
A brief history of NLP (at Penn)

1945: ENIAC finishes development at Moore.
1947: Penn starts first linguistics department in the US.
1951: Noam Chomsky graduates from Penn.
1989: The Treebank dataset is released. (Mitchell P. Marcus, Beatrice Santorini at Penn)
2018: You take this class. (time to contribute!)
NLP Tasks:

Case study: parts of speech tagging

Given: sentence
Do: map each word in sentence to a part of speech
(e.g. noun, verb, verb phrase)
Case study: named entity recognition

Given: sentence
Do: identify and label each named entity in sentence
Case study: intent classification

Given: sentence
Do: map sentence to an intent (set of utterances)
Case study: natural language generation

**Given:** ?
**Do:** create a string *sentence* that has some property
(e.g. is a valid natural language)
NLP Tasks:

Case study: **machine translation**

**Given:** sentence

**Do:** create another natural language string

*translated_sentence* that is in another language
Challenge: **syntax**

Case study: **grammar rules**

**Me:** Dad, I’m hungry.

**Dad:** Hi, hungry.
Challenge: **syntax**

**Case study: buffalo**

**Me:**

**Dad:** Buffalo buffalo Buffalo buffalo buffalo buffalo buffalo Buffalo buffalo.
Challenge: semantics

Case study: homonyms

Me: There is a bug in my code.
Dad: Don’t kill it inside the house!
Challenge: semantics

Case study: sarcasm (is your dad making a joke or not?)

Me: Dad, I’m hungry.
Dad: Hi, hungry.
A Bag of NLP Tricks

Converting to lowercase: (it’s important!)

Tokenization: transform sentence into list of words

Lemmatization: transform word into it’s “root”

N-Gram: split words into n-tuples
insert

machine learning

magic here
Common Machine Learning Models

**Naive Bayes:** probabilistically classifying text

**Markov Models:** probabilistically model language

**Neural Networks:** can solve long term dependencies
How does **machine learning** achieve such *impressive* results?

(*hint:* we don’t really know)
Language Representations

**Strings**: good for individual words/sentences

**Emoji**: good for more semantic power than words

**One Hot Encoding**: good for small corpus/frequencies

**Word Embeddings**: good for lots of things!
Main idea: computer understand numbers, not words.

For every word in the corpus, we want to assign a unique binary value.

How can we implement this?
Word embeddings:

Main idea: we can place words in continuous vector space.

Ideal scenario: we have similar words clustered near each other in vector space. We can solve the homonym problem by computing the euclidean distance.

How do we achieve this?
insert

machine learning

magic here
DISTRIBUTIONAL HYPOTHESIS
Main idea: predicting words that appear near other words (thank you Google!)

Implementation: Word2Vec (Mikolov et. al)

Continuous Bag of Words:
Given center word, predict its neighbours.

Skip Gram:
Given a sentence with a missing word, predict the word that is missing.
Word embedding applications:

1. **Recommendation** *(can we generalize embeddings?)*
2. **Classification** *(can we use word embeddings as neural network inputs?)*
3. **Generation** *(can we construct language from its vector representation?)*
Could we implement these tasks?

Yeah, but we’re lazy. Let’s just import it.
Introducing: spaCy

- Created by Explosion AI
- Collection of models for NLP tasks
- Easy integration with deep learning frameworks
- Supports multiple languages
- Very fast (industry tested)