CIS 192: Python Programming

Natural Language Processing

Arun Kirubaranjan
(kiruba@seas)
@kirubaranajan
kirubaranjan.com
Goal: chatbots

a “holy grail” of computer science est. 1950
it’s a necessary condition (i.e. AI implies understanding natural language)
Problem:
How would you build a chatbot that understands Python?
Solution:
Just run `python` in the terminal.

Python really *is* easy!
Motivation:
can we build a chatbot that understands English?
Let’s build a chatbot!
on the whiteboard (pseudocode is close enough to Python)
What is natural language?

How do humans learn natural language?
How do computers learn natural language?
Intuitively, 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*
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)
NLP Tasks:

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)
NLP Tasks:

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 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
A Wormhole of NLP Tricks

Naive Bayes: probabilistically classifying text
Word Embeddings: encoding lexical semantics
Neural Networks: can do lots of stuff!
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 entire corpus/frequencies

**Word Embeddings:** good for lots of things!
One hot encoding:

**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
Implementation: **Word2Vec** (Mikolov et. al)

**Main idea:** predicting words that appear other words (thank you Google!)

**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?)
Quiz time!

1. What is the difference between syntax and semantics?
2. What technique is the SOTA in sentiment classification?
3. Why are word embeddings more efficient than OHE?
Let’s build a chatbot! (again)

Using some new NLP techniques!
The End!

For now - learn about Recurrent Neural Networks next week!