UF Data Science and Informatics

Python for NLP

Presented By: Tristan Pank

Sign In

Workshop/Event Attendance

Membership Form

Agenda

01 Colab Access

02 Language as Data

03 Frequencies/Inverted Index

04 Capturing “Meaning”

05 Embeddings

06 LLM’s/Transformers

Colab Notebook

https://colab.research.google.com/drive/1P4y008HeV2fMuXyr60DDH83jN99hNbay?usp=sharing

Language as Data

- How can we represent language in a way that machines understand?
  - ASCII: Map each character to a number
    - ‘a’ -> 97
    - ‘A’ -> 65
  - Splitting sentences into tokens
    - “The quick brown fox jumps” -> “The”, “quick”, “brown”, “fox”, “jumps”
- Any issues?
  - How should a machine interpret a word?
  - 26 unique characters in English, but magnitudes more words
  - How do we distinguish between the “meaning” of two words. “Love” vs “Hate”

Basic NLP – Searching

- Goal: How can we use machines to help search through documents?
- Inverted Index:
  - Split documents and query into tokens
  - Return the document matching the most words to the query
  - Ex: “How to study machine learning?”

Documents	Matched Keywords	Score
“How to study biology”	How, to, study	3
“Guide for machine learning”	machine learning	2

- What’s wrong with this approach?
  - While “perfect” queries work, replacing words with synonyms leads to different results
  - “study” and “learn” can mean the same thing, but have different uses in inverted index
  - We need an approach that can capture the meaning of words

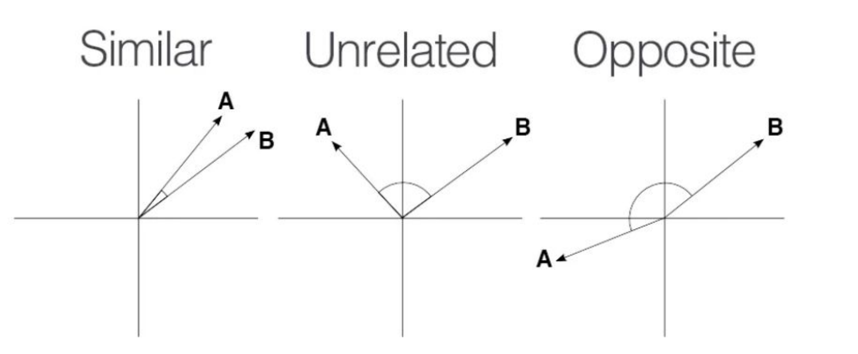
Basic NLP - Pre-Processing

- How can we make NLP models more accurate?
- Pre-Process prior to inputting data into a model
  - Lowercase each token
  - Remove punctuation
  - Remove stopwords (“the”, “is”, “in”)
  - Lemmatization
    - Reduces a word to its base form
    - “run”, “running”, “runs”, “ran” are all different words but essentially mean the same thing
    - Models can get confused and think they are different
    - Lemmatization reduces all of these to “run”

Embeddings

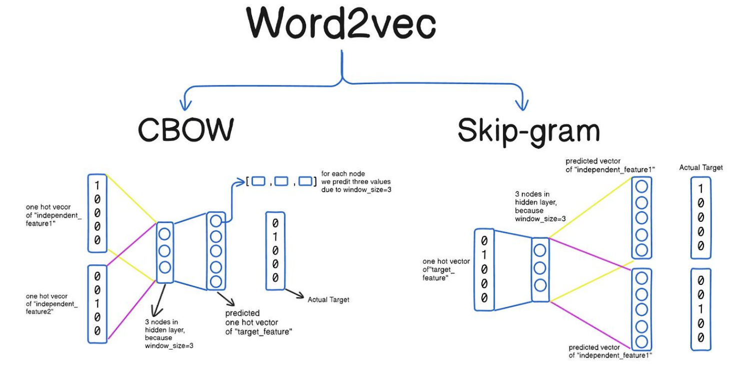
- Goal: Represent words with numbers in a meaningful way
- Cannot just assign each word a number
    - Too many words (100,000+)
    - Too difficult to group similar words
    - Only way to compare is distance
- Vectors
    - Encode each word into a vector of a higher dimensional space
    - Vectors allow for other forms of comparison besides distance
    - Meaning is captured in the latent space itself
    - Two words are “similar” if they have a similar direction
    - Cosine Similarity measures the angle between two vectors. Ranges from [-1,1].

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$



Embeddings – How to?

- Co-Occurrence Matrix:
    - Given a word count how often other words are next to it
    - Assemble a vector for a word with each element being the count
- Neural Networks (Word2Vec):
    - Input -> Hidden/Embedding Layer -> Output (Softmax)
    - Skip-Gram
        - Given a word (e.g “cat”), try to predict the words prior and after it (e.g “the”,”sat” -> “the cat sat”)
        - After training, the hidden layer should provide a sufficient vector representation of the word
    - Continuous Bag of Words
        - Given vectors of surrounding words predict the center word.



https://medium.com/@fraidoonomarzai99/word2vec-cbow-skip-gram-in-depth-88d9cc340a50

Transformers

- An encoder-decoder model that utilizes self-attention for fast training and accurate results
- Attention:
  - Instead of processing one word at a time, attention can process entire sentences
  - Each token is embedded originally, and then adjusted based on surrounding words to capture context
  - Results in better embeddings that not only capture words, but meaning.
  - “bank” in “river bank” and “money bank” mean different things
  - Attention first encodes “bank” to a vector, then adjusts the vector based on surrounding words
- Encoders create and return these embeddings: BERT, Sentence Transformers, etc.
- Decoders generate new words from the embedding space with probabilistic methods: LLM’s like Chat GPT