AI SATURDAYS LAGOS

Free Text and Natural Language Processing

by

Wuraola Oyewusi

What is Natural Language Processing?

Natural language processing (NLP) is a collective term referring to automatic computational processing of human languages. This includes both algorithms that take human-produced text as input, and algorithms that produce natural looking text as outputs.

— Page xvii, Neural Network Methods in Natural Language Processing, 2017.

What is Natural Language Processing?

Natural language processing (NLP) is a subfield of linguistics, computer science information engineering, and artificial intelligence concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data.

Challenges in natural language processing frequently involve speech recognition, natural language understanding, and natural language generation.

-Wikipedia, 2020

What is Natural Language Processing?

Natural language processing (NLP) is an interdisciplinary subfield of computer science and linguistics. It is primarily concerned with giving computers the ability to support and manipulate speech. It involves processing natural language datasets, such as text corpora or speech corpora, using either rule-based or probabilistic (i.e. statistical and, most recently, neural network-based) machine learning approaches. The goal is a computer capable of "understanding" the contents of documents, including the contextual nuances of the language within them. The technology can then accurately extract information and insights contained in the documents as well as categorize and organize the documents themselves.

-Wikipedia September 2023

What is Free Text ?

Text data usually consists of documents which can represent words, sentences or even paragraphs of free flowing text.
The inherent unstructured (no neatly formatted data columns!) and noisy nature of textual data makes it harder for machine learning methods to directly work on raw text data.

Dipanjan (DJ) Sarkar, January 2019

How big is text data on the Internet

- Let's find out together (Stop sharing screen to login)
- https://chat.openai.com/

Applications of Natural Language Processing

- Natural language generation
- Speech recognition
- Speech synthesis
- Ontology population
- Question answering
- Machine translation
- Search engine
- Text coherence
- Fake news detection

https://natural-language-understanding.fandom.com/wiki/List_of_natural_language_processing_tasks

Tasks in Natural Language Processing

- Tokenization
- Sentence boundary detection
- Shallow parsing
    - Part-of-speech tagging
    - Selectional preference
    - Noun phrase chunking
- Syntax parsing
    - Dependency parsing
    - Constituency parsing

https://natural-language-understanding.fandom.com/wiki/List_of_natural_language_processing_tasks

Tasks in Natural Language Processing

- Semantics
  - Semantic role labeling
  - Spatial role labeling
  - Semantic dependency parsing
- Pragmatics
- Sentiment analysis / opinion mining
- Word sense disambiguation/induction

https://natural-language-understanding.fandom.com/wiki/List_of_natural_language_processing_tasks

Tasks in Natural Language Processing

- Named-entity
  - Named-entity recognition/classification
  - Entity linking
  - Temporal expression recognition/normalization
  - Co-reference resolution
- Information extraction
- Terminology extraction
- Discourse parsing
- Topic modeling

https://natural-language-understanding.fandom.com/wiki/List_of_natural_language_processing_tasks

Tasks in Natural Language Processing

- Summarizing
- Similarity
  - Attributional similarity (word similarity)
  - Relational similarity
  - Phrase similarity
  - Sentence similarity
  - Paraphrase identification
  - Textual entailment

https://natural-language-understanding.fandom.com/wiki/List_of_natural_language_processing_tasks

Can computers directly process text data?

Raw Text → Feature Representation → ML Algorithms

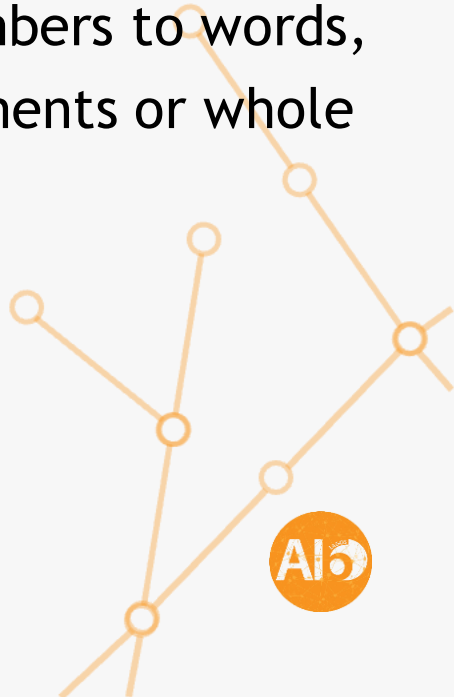
Can computers directly process text data?

The classifiers and learning algorithms cannot directly process the text documents in their original form, as most of them expect numerical feature vectors with a fixed size rather than the raw text documents with variable length.

Can computers directly process text data?

The process of transforming text into numeric stuff, is usually performed by building a language model. These models typically assign probabilities, frequencies or some obscure numbers to words, sequences of words, group of words, section of documents or whole documents

Michel Kana, July 15,2018



Let's do some Alphabet Encoding Together

A	B	C	D	E	F	G	H	I	J	K	L	M
1	2	3	4	5	6	7	8	9	10	11	12	13

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
14	15	16	17	18	19	20	21	22	23	24	25	26

Techniques for Text representation

- One-Hot Encoding
- N-Grams
- Bag-of-words
- Term Frequency-Inverse Document Frequency (TF-IDF)
- Word Embeddings
- Transformers

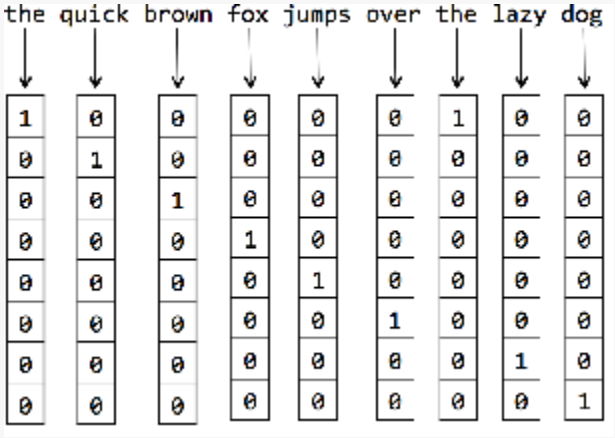
One-Hot Encoding

A one hot encoding is a representation of categorical variables/tokens as binary vectors.

This first requires that the categorical values be mapped to integer values.

Then, each integer value is represented as a binary vector that is all zero values except the index of the integer, which is marked with a 1.

One-Hot Encoding



Source: Fundamentals of Deep Learning, N. Buduma, 2017

One-Hot Encoding

Pros

- Simple
- Independent

Cons

- Ineffective for Large Vocabularies
- Misses the relationship between words
- Computation inefficiency

N-grams

N-grams are continuous sequences of n items, which can be characters, words, or other linguistic units, extracted from a text'.
The longer the context on which we train a N-gram model, the more coherent the sentences we can generate.

Furthermore, the N-gram model is heavily dependent on the training corpus used to calculate the probabilities. One implication of this is that the probabilities often encode specific facts about a given training text, which may not necessarily apply to a new text

N-grams

N-gram language models estimate the probability of the last words given the previous words. It finds use in spell checking, auto completion, language identification ,text generation etc.

Sentence = "Welcome to AI Saturday Lagos"

1-gram(or unigram) : "Welcome", "to", "AI", "Saturday", "Lagos"

2-gram (or bigram) : "Welcome to", "to AI" ."AI Saturday"," Saturday Lagos"

3-gram(or trigram) : “Welcome to AI" , AI Saturday Lagos"

N grams

Pros

- Local context and Sequential Information
- Variable length to capture different length of dependencies

Cons

- Sparse data
- High dimensionality
- Sensitive to order
- Limited semantic capture

Bag of Words

For tasks that are not based on sequential pattern of words maybe like classifying texts based on sentiments or detecting the language a text is written in.

Texts can be represented by bag of words, ignoring their original position in the text, keep only their frequency.
This method relies on term frequency, the number of times a token shows up in a document is counted and this value is used as its weight

Bag of Words

Pros

- Simple
- Efficient especially with very large datasets

Cons

- Loss of Sequence order
- High dimensionality
- Fixed length representation

Term Frequency-Inverse Document Frequency (TF-IDF)

TF-IDF stands for “term frequency-inverse document frequency”, meaning the weight assigned to each
token not only depends on its frequency in a document but also how recurrent that term is in the entire corpora.

Term Frequency :

Term frequency = (Number of Occurrences of a word)/(Total words in the document)

Inverse document frequency:

IDF(word) = Log((Total number of documents)/(Number of documents containing the word))

Term Frequency-Inverse Document Frequency (TF-IDF)

Pros

- Weighs word relevance
- Penalizes common words

Cons

- Ignores Sequence order
- Limited semantic understanding
- Requires document level information

Word Embeddings

Word embeddings are dense vector representations of words in a continuous vector space.

They capture both semantic and contextual meaning of words.
E.g The bank on the bank of the river

Examples of Word Embedding

Word2Vec
GloVe
FastText

Word embeddings

Pros

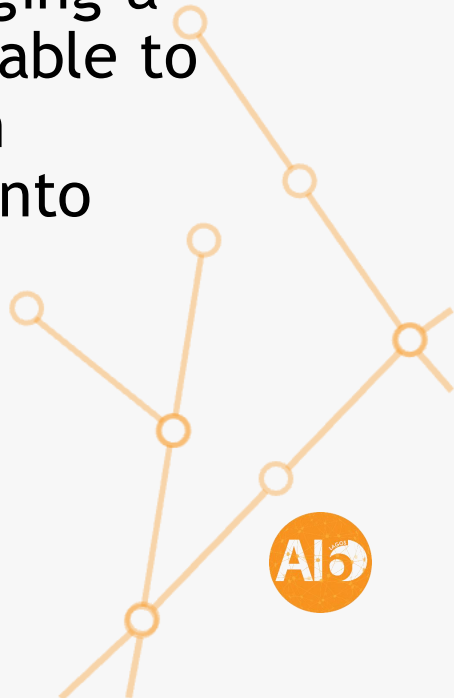
- Semantic and Contextual Representation
- Pre trained embeddings

Cons

- Out of Vocabulary words
- Fixed Context

Transformers

Transformers are able to represent text leveraging a concept called attention mechanism, they are able to capture long term dependencies by focusing on different parts of input text when encoding it into vectors, they also capture context effectively



Word embeddings

Pros

- State of the Art Performance(SOTA)
- Multimodal Application
- Transfer Learning
- Multilingual support

Cons

- Computational Intense
- Overfitting and Hallucination

General Text Preprocessing

- Remove HTML tags
- Remove extra whitespaces
- Convert accented characters to ASCII characters
- Expand contractions
- Remove special characters
- Lowercase all texts
- Convert number words to numeric form
- Remove numbers
- Remove stopwords
- Lemmatization

https://www.kdnuggets.com/2018/08/practitioners-guide-processing-understanding-text-2.html

Thank You