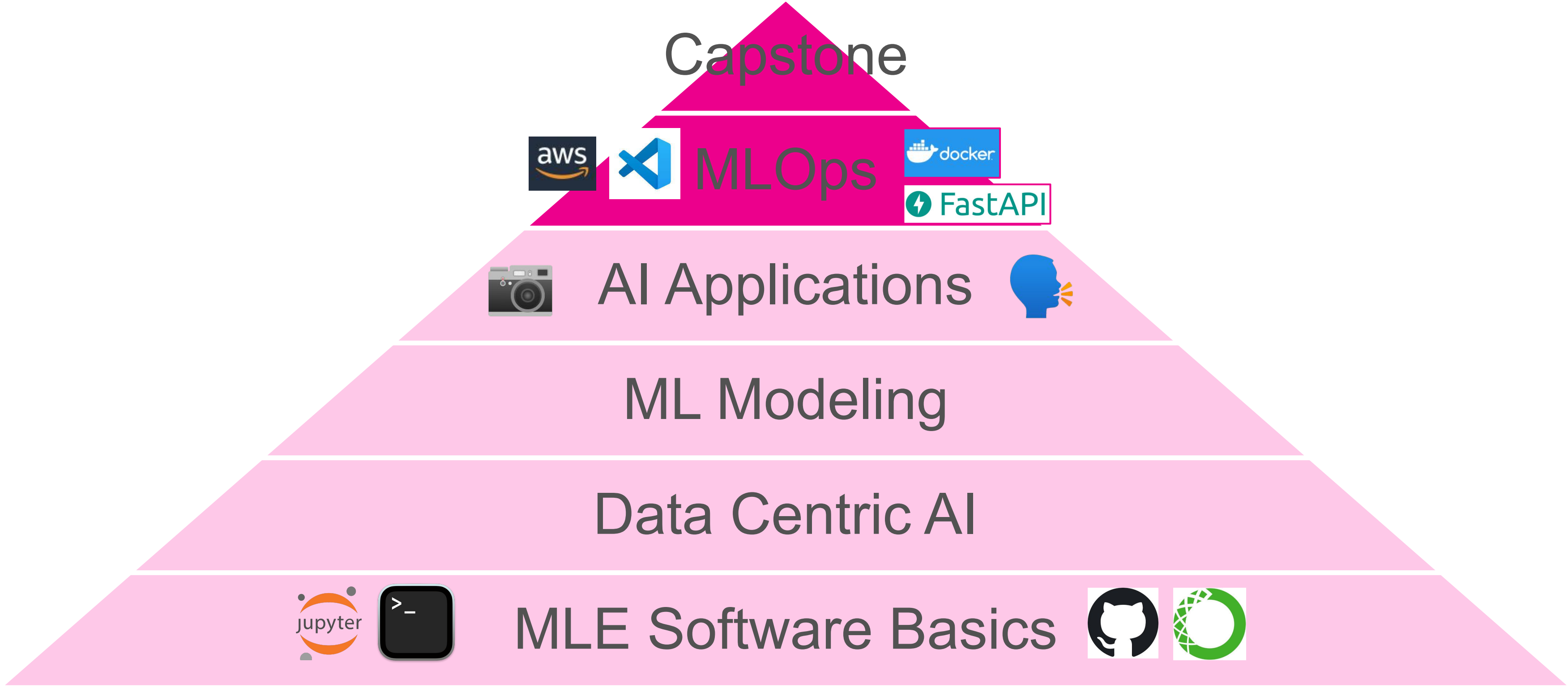
FourthBrain

MLE Program, Cohort 11 (MLE11)

Week 10: Natural Language Benchmarks, Dealing with Text, NER, Feature Extraction, Word Embedding

Becoming a Machine Learning Engineer



Capstone

MLOps

AI Applications

ML Modeling

Data Centric AI

MLE Software Basics

Our Updated Curriculum!

1. ML Project Scoping
2. Real, Live Data Streams
3. Data Wrangling & Exploratory Analysis
4. Big Data

DATA CENTRIC AI

5. Supervised ML
6. Deep Learning & AutoML
7. Unsupervised, Semi- & Self-supervised Learning

ML MODELING

8. Computer Vision
9. Natural Language Processing
10. Transformers & Fine Tuning Pre-Trained Networks

AI APPLICATIONS

11. Building ML Web Apps
12. Containerization
13. Model Serving
14. Machine Learning in Production

MLOps

Last Week!

Concepts

- Computer Vision Benchmarks
- Dealing with Images
- Object Detectors
- Semantic Segmentation
- Explainability & Saliency

Hands on

- Predicting pathology to automate patient prioritization

🤖 This Week!

Concepts

- Natural Language Benchmarks
- Dealing with Text
- Named Entity Recognition
- Bag of Words, Term Frequency Inverse Document Frequency
- Tokenization & Word Embeddings

Hands on

- Named Entity Recognition from firm financial documents

What questions do you have?

Reminders To-Do

- Prepare 10 min Capstone Presentation
    - focus on progress so far
    - roadblocks and revised timeline
    - include deployment plans

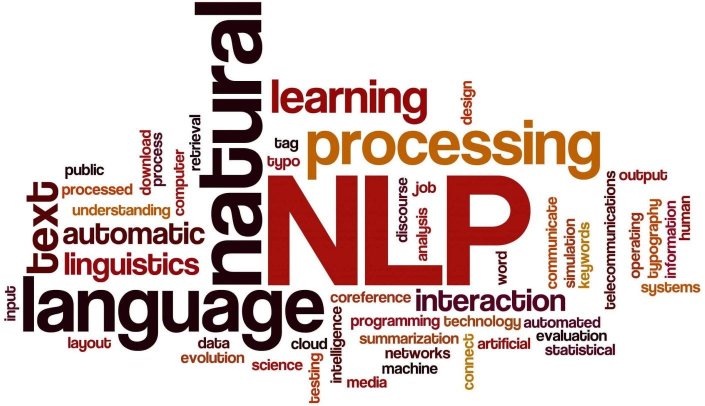
Natural Language Processing

What is NLP?

- NLP – Natural Language Processing
- NLP refers to the branch of computer science, and AI, concerned with giving computers the ability to understand text and spoken words in much the same way human beings can
- It combines computational linguistics – rule-based modeling of human language with statistical, machine learning, and deep learning models
- NLP drives computer programs that translate text from one language to another

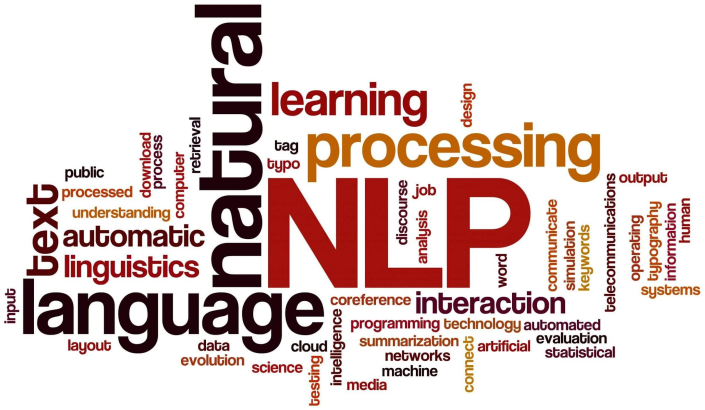
Where is NLP used?

- Machine translation
- Response to spoken commands
- Text summarization
- Voice-operated systems
- Digital assistants
- Speech-to-text dictation
- Chatbots
- Spam detection



NLP tasks

- Speech Recognition
- Sentences tokenization
- Part of Speech tagging
- Word sense disambiguation
- Named Entity Recognition
- Co-reference resolution
- Sentiment Analysis
- Natural Language generation



Dealing with Text

Dealing with text

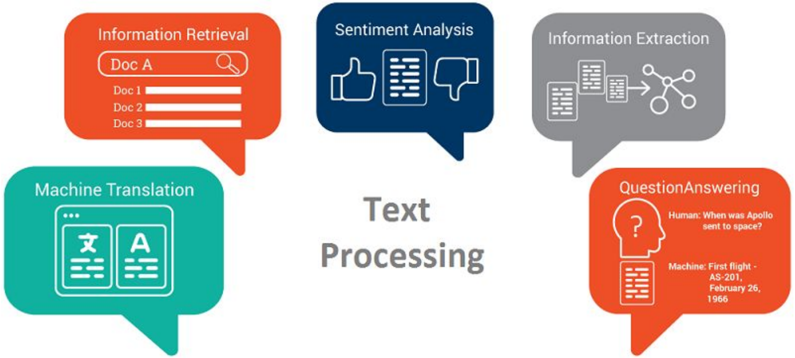
- To achieve the level of automation and prediction needed, the machine needs to understand the input textual data

- Unfortunately, this understanding is not as easy to achieve as human reading text

- The data needs to be cleaned, modeled, and transformed in multiple manners so that it could be used by the machines

Text Processing

- Text processing is the practice of automating the creation or manipulation of electronic text
- Text usually refers to all the alphanumeric characters specified on the keyboard. But in general, “text” means the abstraction layer, immediately above the standard character encoding of the target text



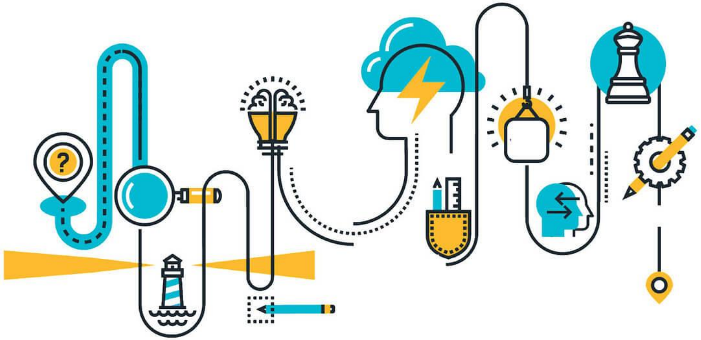
Text Processing

- Text processing takes the raw input text, clean it, normalize it, and converts it into a form that is suitable for feature extraction

- Extracting plain text:
    - Textual data can come from a wide variety of sources:
        - Web
        - PDFs
        - Word Document
        - Speech Recognition systems
        - Book Scans etc.

Text Processing | Journey

- The following process is not linear, you might go back and forth several times and might require additional steps:
  - Cleaning
  - Normalizing
  - Tokenization
  - Stop Word Removal
  - Part of Speech Tagging
  - Named Entity Recognition
  - Stemming and Lemmatization
  - Feature Extraction
  - Modelling



Text Processing | Cleaning

- The cleaning step consists of removing irrelevant items, such as HTML tags
- The *requests* library in python helps fetch a webpage
- Then you can use *BeautifulSoup* library to extract the plain text

(this library takes care automatically of the nested tags)

Checkout this tutorial!

Text Processing | Normalization

- Normalization usually Includes 2 steps

- Case Normalization: Converting all the text to lowercase
    - This could be done using the *.lower()* function in python

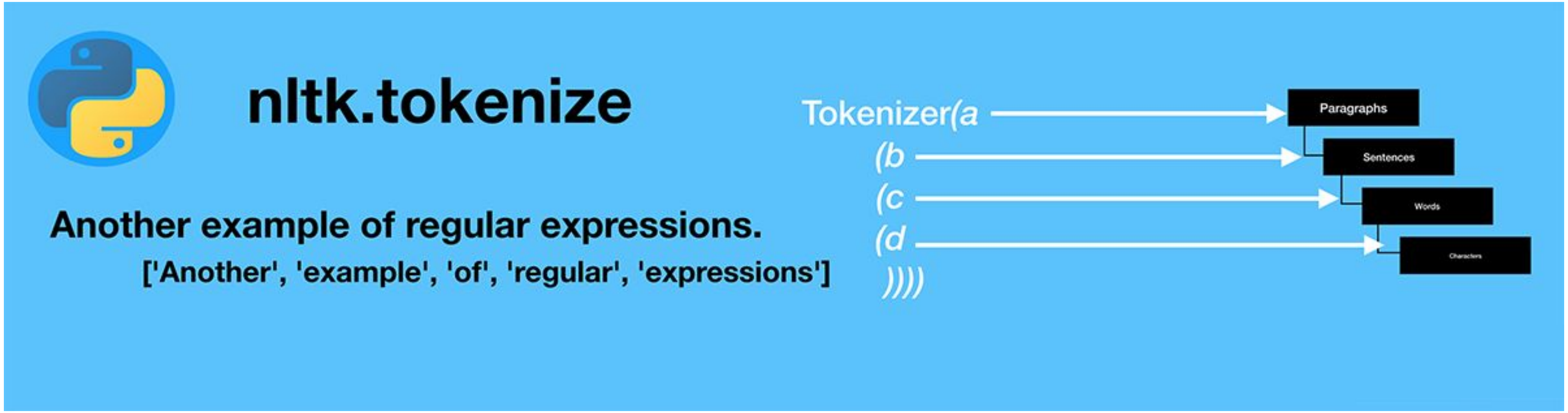
- Removing Punctuation
    - This could be achieved using the *re* library to remove punctuation with a regular expression (regex)

Text Processing | Tokenization

- This step splits the text into words or **tokens** (symbols)
- It could also refer to splitting each sentence into a sequence of words
- Natural Language Toolkit *nltk.tokenize* package is used to help tokenize into sentences and words.
- NLTK is also considered as a tweet handler that handles hashtags and emoticons

Text Processing | Tokenization

- Word Embedding is a concept that relies heavily on tokenization

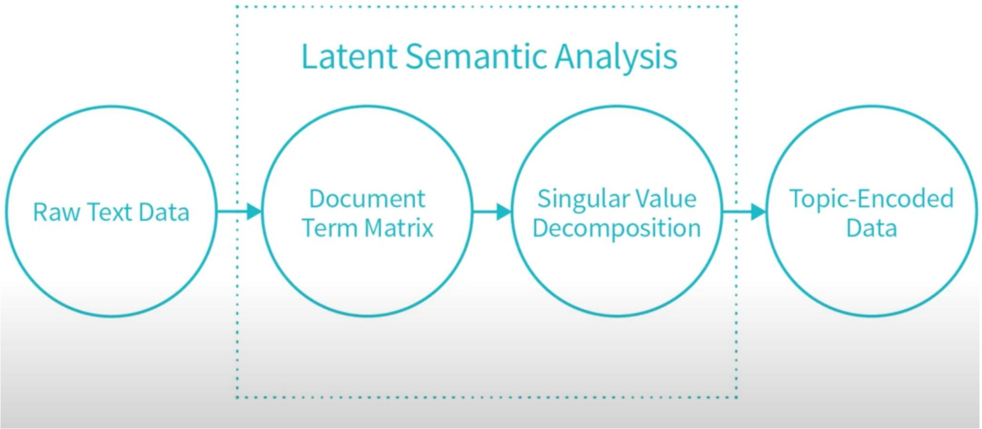


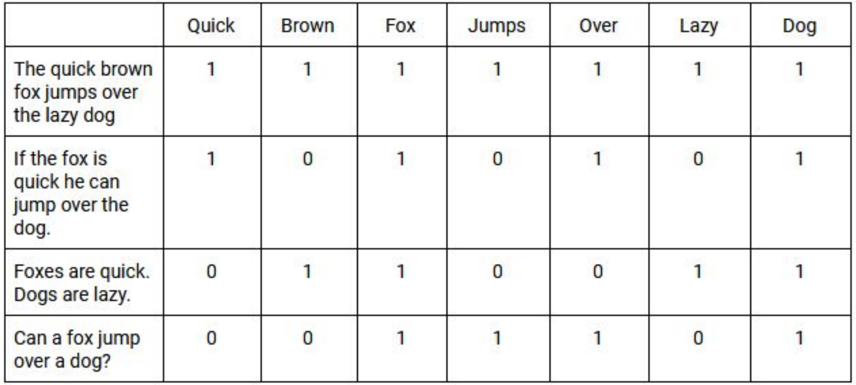
Another example of regular expressions.

['Another', 'example', 'of', 'regular', 'expressions']

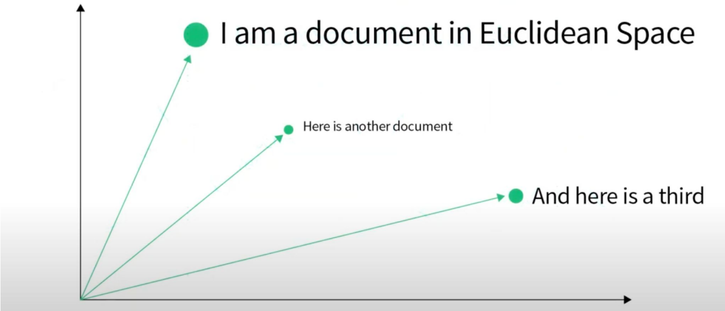
Tokenization and Word Embedding

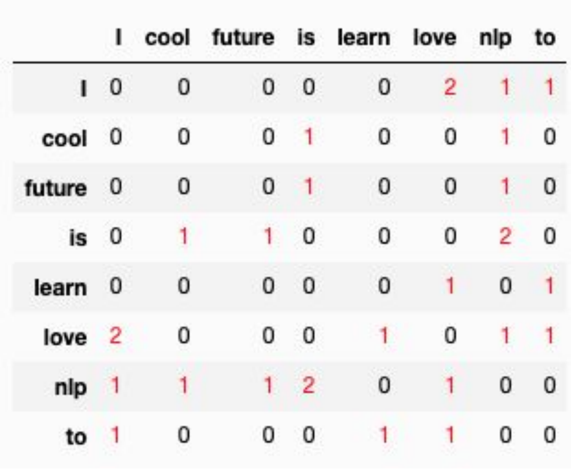
Latent Semantic Analysis (LSA)





	Quick	Brown	Fox	Jumps	Over	Lazy	Dog
The quick brown fox jumps over the lazy dog	1	1	1	1	1	1	1
If the fox is quick he can jump over the dog.	1	0	1	0	1	0	1
Foxes are quick. Dogs are lazy.	0	1	1	0	0	1	1
Can a fox jump over a dog?	0	0	1	1	1	0	1





	I	cool	future	is	learn	love	nlp	to
I	0	0	0	0	0	2	1	1
cool	0	0	0	1	0	0	1	0
future	0	0	0	1	0	0	1	0
is	0	1	1	0	0	0	2	0
learn	0	0	0	0	0	1	0	1
love	2	0	0	0	1	0	1	1
nlp	1	1	1	2	0	1	0	0
to	1	0	0	0	1	1	0	0

Word Embedding

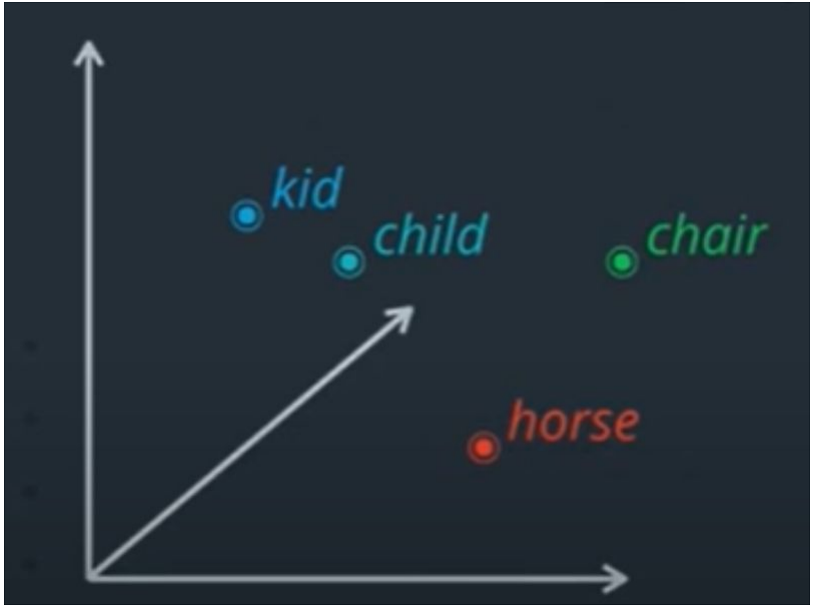
- Word embeddings are a type of word representation that allows words with similar meanings to have a similar representation

- It controls the size of the word representation by limiting it to a fixed-size vector

- It is considered as one of the key breakthroughs of deep learning on challenging natural language processing problems

Word Embedding | Example

- If 2 words are similar in meaning, they should be closer to each other compared to words that are not



Word Embedding | Benefits

- The majority of neural network toolkits do not play well with very high-dimensional, sparse vectors

- The main benefit of the dense representation that word embeddings achieve is the generalization power

- If we believe that some features may provide similar clues, it is worthwhile to provide a representation that is able to capture these similarities

Word Embedding | Embedding Layer

- An embedding layer, is a word embedding that is learned jointly with a neural network model on a specific natural language processing task, such as language modeling, or document classification

- It requires that the document text is cleaned and prepared such that each word is one-hot-encoded.

- The vectors are initialized with small random numbers

Word Embedding | Embedding Layer

- The embedding layer is used on the front end of a neural network and is fit in a supervised way using Backpropagation Algorithm

- The one-hot-encoded words are mapped to the word vectors

- If a multilayer Perceptron model is used, then the word vectors are concatenated before being fed as an input to the model

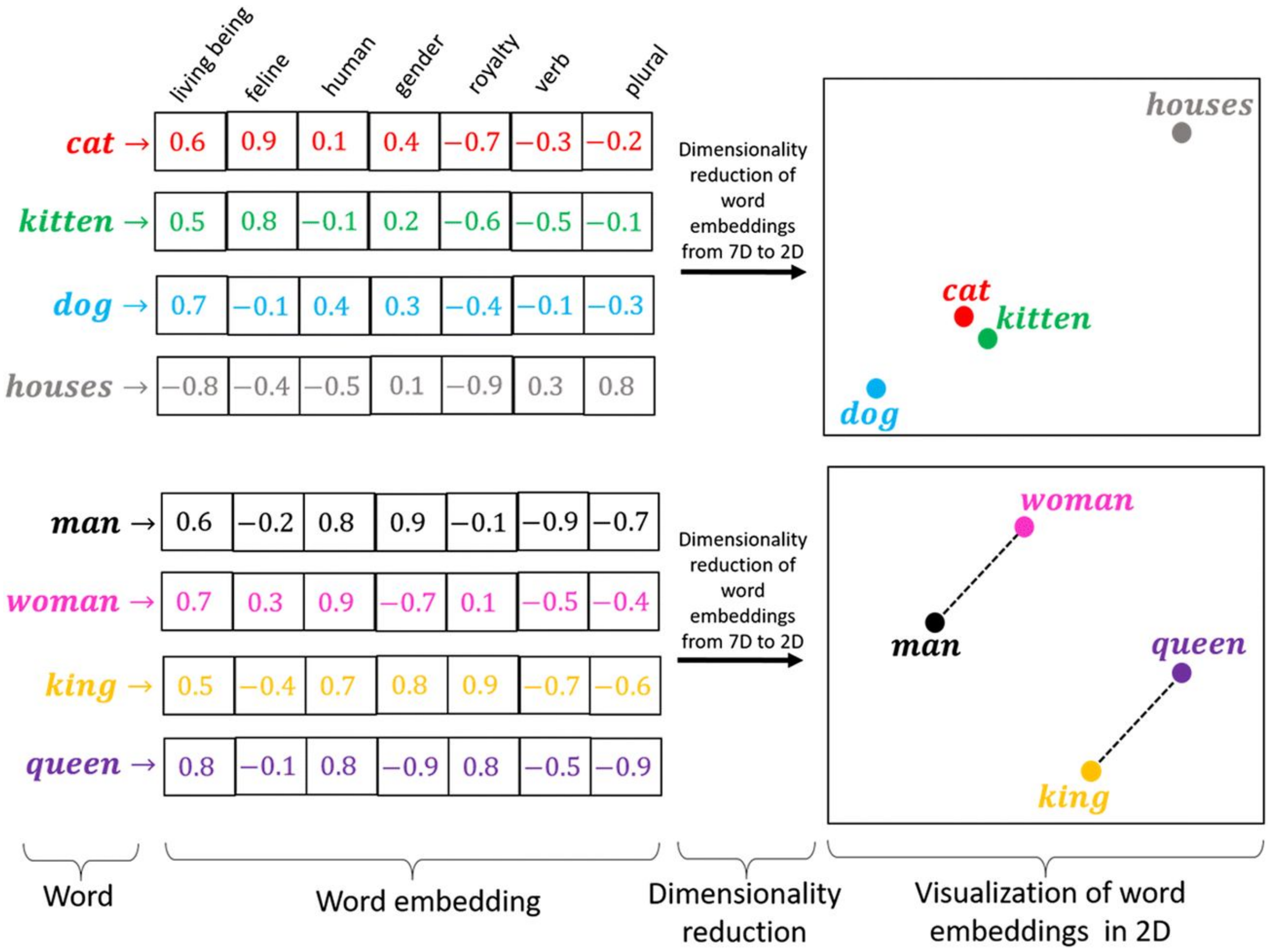
- If a recurrent neural network is used, then each word may be taken as one input in a sequence

Word Embedding | Embedding Layer

- By combining word vectors, we can come up with another way of representing documents

- This method helps in:
  - Finding Synonyms and analogies
  - Identifying concepts around which words are clustered
  - Classifying words as positive, negative, or neutral

- This approach requires a lot of training data and can be slow, but will learn an embedding both targeted to the specific text data and NLP task



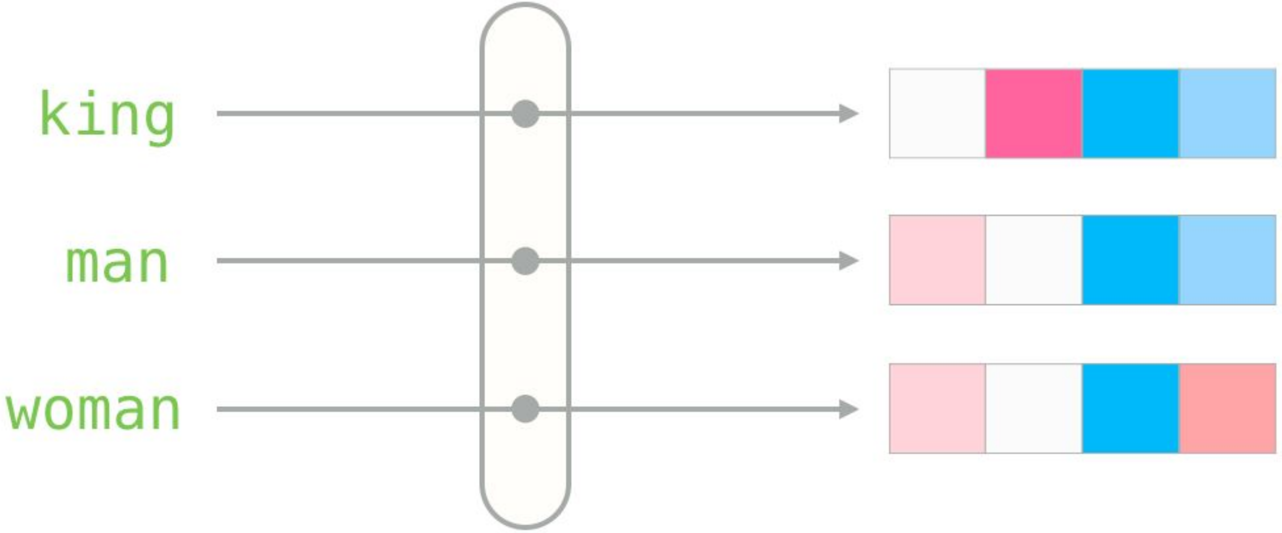
	living being	feline	human	gender	royalty	verb	plural
cat →	0.6	0.9	0.1	0.4	−0.7	−0.3	−0.2
kitten →	0.5	0.8	−0.1	0.2	−0.6	−0.5	−0.1
dog →	0.7	−0.1	0.4	0.3	−0.4	−0.1	−0.3
houses →	−0.8	−0.4	−0.5	0.1	−0.9	0.3	0.8

man →	0.6	−0.2	0.8	0.9	−0.1	−0.9	−0.7
woman →	0.7	0.3	0.9	−0.7	0.1	−0.5	−0.4
king →	0.5	−0.4	0.7	0.8	0.9	−0.7	−0.6
queen →	0.8	−0.1	0.8	−0.9	0.8	−0.5	−0.9

Dimensionality reduction of word embeddings from 7D to 2D
houses
cat
kitten
dog
Dimensionality reduction of word embeddings from 7D to 2D
woman
man
queen
king
Word
Word embedding
Dimensionality reduction
Visualization of word embeddings in 2D

source: https://medium.com/@hari4om/word-embedding-d816f643140

Word2Vec



Word2Vec

- Word2vec is a method for learning continuous-valued word embeddings from large data sets. It was developed by a team at Google led by Tomas Mikolov in 2013.

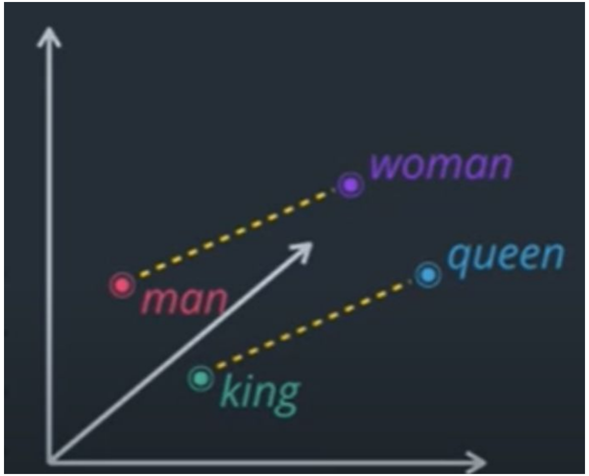
- It came as a response to make the neural-network-based training of the embedding more efficient and since then has become the de facto standard for developing pre-trained word embeddings.

Word2Vec

- The work involved analysis of the learned vectors and exploration of vector math on the representations of words.

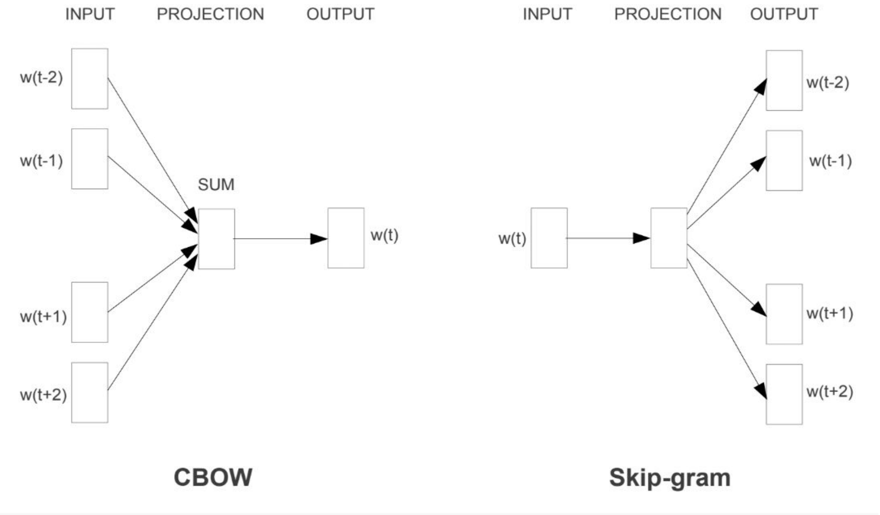
- Example:
    - Subtracting the “man-ness” from “king” and adding “women-ness” results in the word “Queen”
    - This captures the analogy that:

“King is to queen as man is to woman”



Word2Vec | learning models

- There are 2 different learning that can be used as part of the word2vec approach to learn the word embedding:
    - Continuous Bag-of-Words (CBOW) model
    - Continuous Skip-Gram model



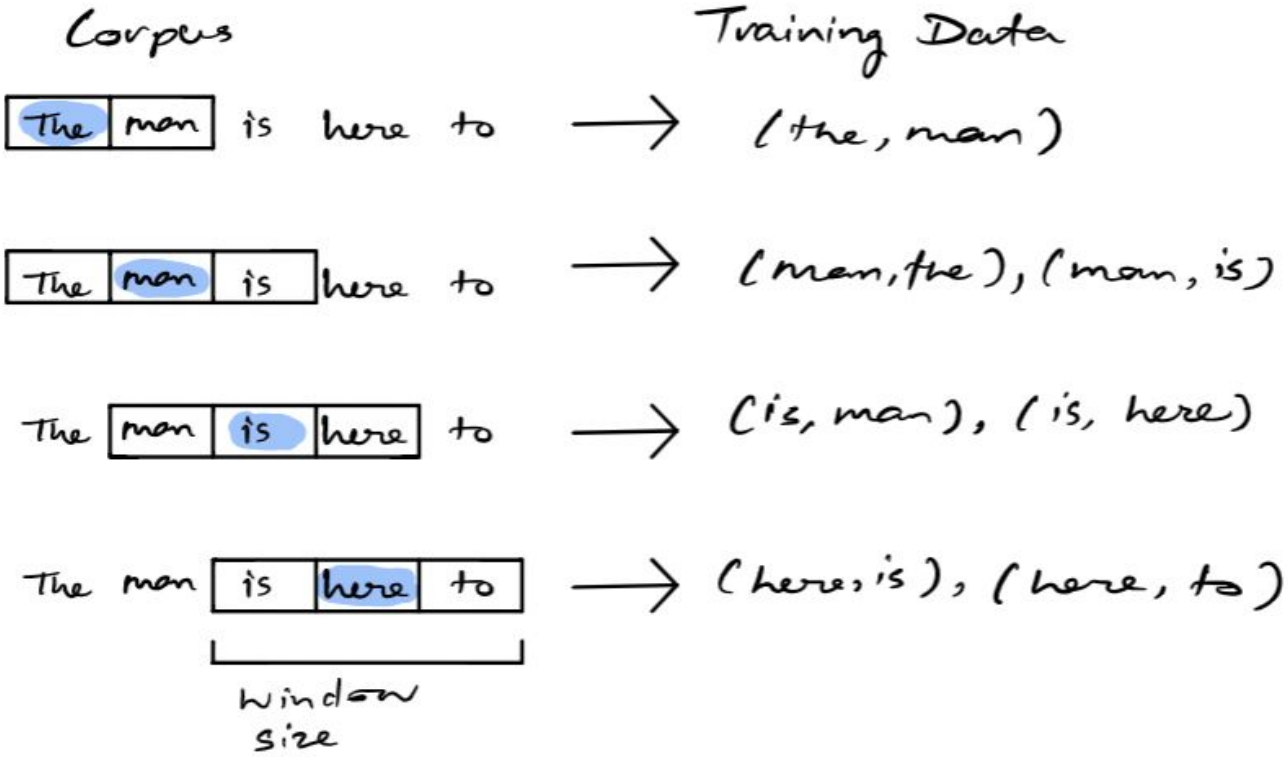
Word2Vec | learning models

- The CBOW model learns the embedding by predicting the current word based on its context

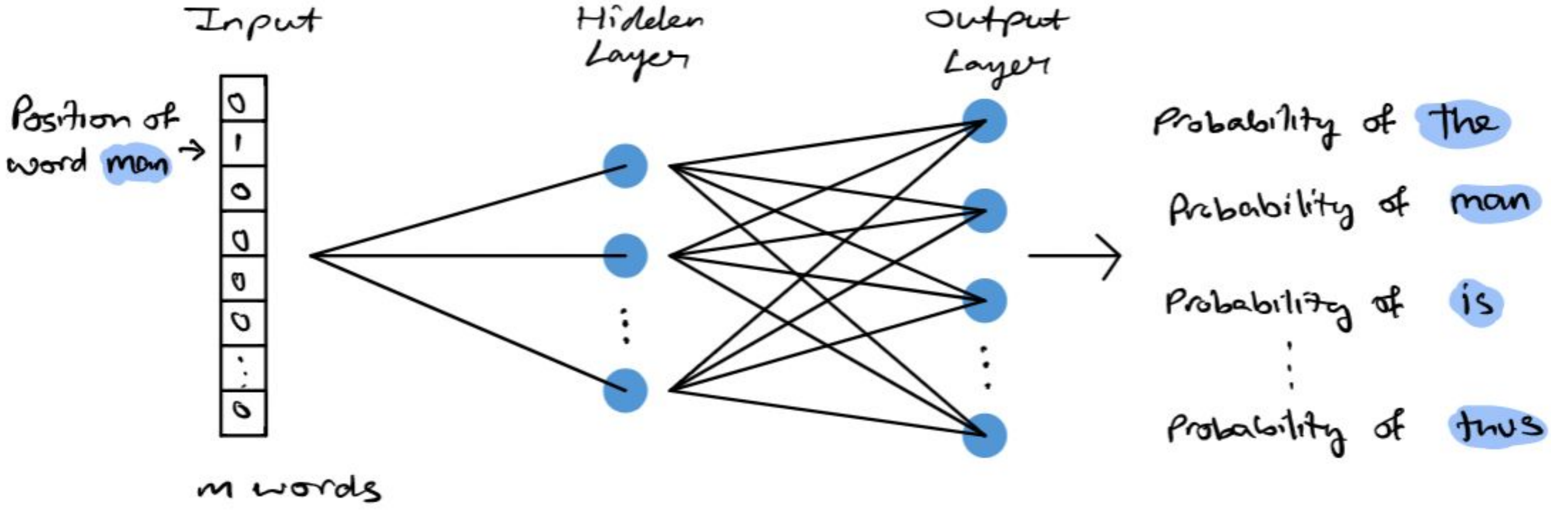
- The continuous skip-gram model learns by predicting the surrounding words given a current word

- Both models are focused on learning about words given their local usage context, where the context is defined by a window of neighboring words. (This window is a configurable parameter of the model)

Data generation



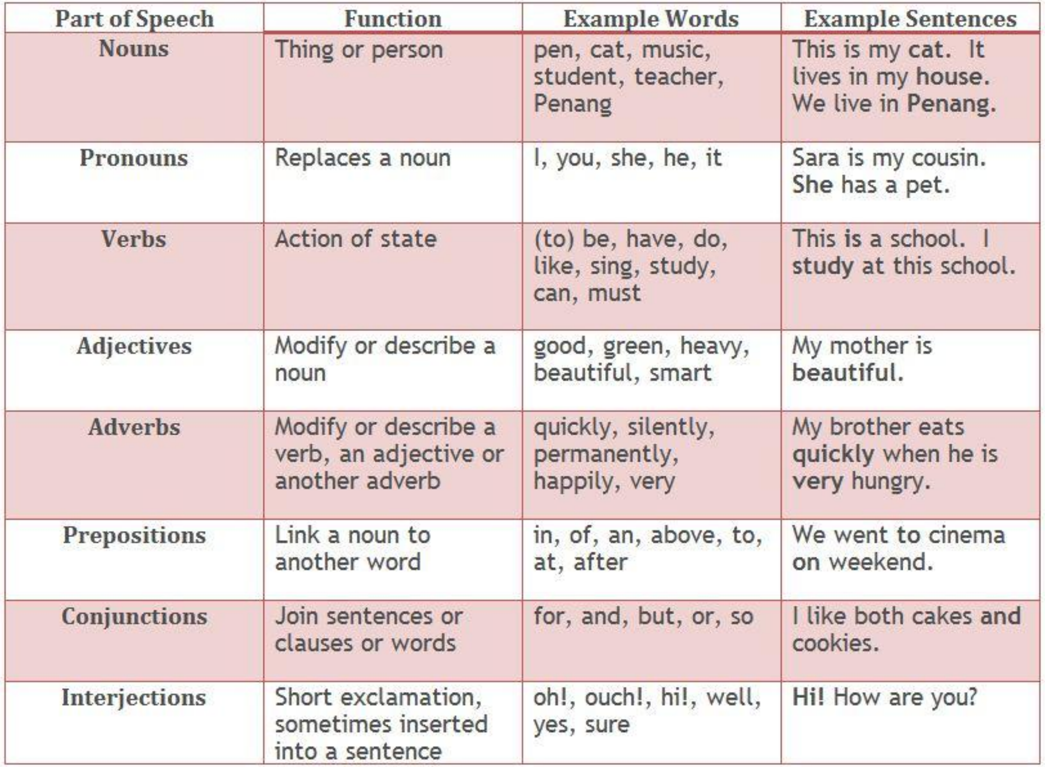
Data generation and Training



Text Processing | Stop Word Removal

- Removing words that are too common
- Removing uninformative words (i.e. is, the, in, at, etc.) that do not add meaning to a sentence
- NLTK is also used to remove stop words using *nltk.corpus.stopwords*

Text Processing | Part of Speech Tagging



Part of Speech	Function	Example Words	Example Sentences
Nouns	Thing or person	pen, cat, music, student, teacher, Penang	This is my **cat**. It lives in my **house**. We live in **Penang**.
Pronouns	Replaces a noun	I, you, she, he, it	Sara is my cousin. **She** has a pet.
Verbs	Action of state	(to) be, have, do, like, sing, study, can, must	This **is** a school. I **study** at this school.
Adjectives	Modify or describe a noun	good, green, heavy, beautiful, smart	My mother is **beautiful**.
Adverbs	Modify or describe a verb, an adjective or another adverb	quickly, silently, permanently, happily, very	My brother eats **quickly** when he is **very** hungry.
Prepositions	Link a noun to another word	in, of, an, above, to, at, after	We went **to** cinema **on** weekend.
Conjunctions	Join sentences or clauses or words	for, and, but, or, so	I like both cakes **and** cookies.
Interjections	Short exclamation, sometimes inserted into a sentence	oh!, ouch!, hi!, well, yes, sure	**Hi!** How are you?

Text Processing | POS

- POS – Part of Speech Tagging
- POS is a popular Natural Language Processing process that refers to categorizing words in a text (corpus) in correspondence with a particular part of speech
- This POS depends on the definition of the word and its context
- NLTK package also helps with POS using python

Text Processing | POS

- Part of speech tags describe the characteristic structure of the lexical terms within a sentence or text

- They are used for making assumptions about the semantics

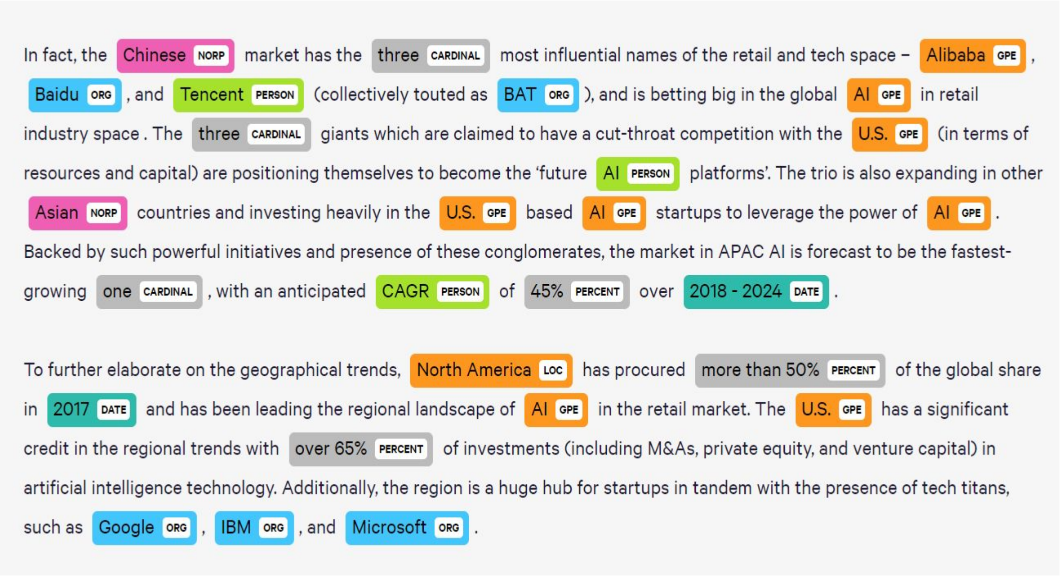
- Other applications of POS tagging include:
    - Named Entity Recognition
    - Co-reference Resolution
    - Speech Recognition

Named Entity Recognition

Text Processing | Named Entity Recognition

The dataset consists of the following tags

- PERSON: People, including fictional.
- NORP: Nationalities or religious or political groups.
- FAC: Buildings, airports, highways, bridges, etc.
- ORG: Companies, agencies, institutions, etc.
- GPE: Countries, cities, states.
- LOC: Non-GPE locations, mountain ranges, bodies of water.
- PRODUCT: Objects, vehicles, foods, etc. (Not services.)
- EVENT: Named hurricanes, battles, wars, sports events, etc.
- WORK_OF_ART: Titles of books, songs, etc.
- LAW: Named documents made into laws.
- LANGUAGE: Any named language.
- DATE: Absolute or relative dates or periods.
- TIME: Times smaller than a day.
- PERCENT: Percentage, including ”%“.
- MONEY: Monetary values, including unit.
- QUANTITY: Measurements, as of weight or distance.
- ORDINAL: “first”, “second”, etc.
- CARDINAL: Numerals that do not fall under another type.



In fact, the Chinese NORP market has the three CARDINAL most influential names of the retail and tech space – Alibaba GPE , Baidu ORG , and Tencent PERSON (collectively touted as BAT ORG), and is betting big in the global AI GPE in retail industry space . The three CARDINAL giants which are claimed to have a cut-throat competition with the U.S. GPE (in terms of resources and capital) are positioning themselves to become the ‘future AI PERSON platforms’. The trio is also expanding in other Asian NORP countries and investing heavily in the U.S. GPE based AI GPE startups to leverage the power of AI GPE . Backed by such powerful initiatives and presence of these conglomerates, the market in APAC AI is forecast to be the fastest-growing one CARDINAL , with an anticipated CAGR PERSON of 45% PERCENT over 2018 - 2024 DATE .

To further elaborate on the geographical trends, North America LOC has procured more than 50% PERCENT of the global share in 2017 DATE and has been leading the regional landscape of AI GPE in the retail market. The U.S. GPE has a significant credit in the regional trends with over 65% PERCENT of investments (including M&As, private equity, and venture capital) in artificial intelligence technology. Additionally, the region is a huge hub for startups in tandem with the presence of tech titans, such as Google ORG , IBM ORG , and Microsoft ORG .

Text Processing | NER

- NER – Named Entity Recognition

- Sometimes referred to as entity chunking, extraction, or identification

- It is the task of identifying and categorizing key information (entities) in text

- An entity can be any word or series of words that consistently refers to the same thing

Text Processing | NER

- Every detected entity is classified into a predetermined category
- For example, “super.AI” could be identified and classified as a “Company”
- NLTK, SpaCy, and Stanford NER are open-source libraries that can help with NER
- NER is a 2-step process:
    - Detect a named entity
    - Categorize the entity

Text Processing | NER

NER – step 1:

- This step involves detecting a word or string of words that form an entity
- Each word represents a token
- “The Great Lakes” is a string of three tokens that represents one entity
- Inside-outside-beginning tagging is a common way of indicating where entities begin and end

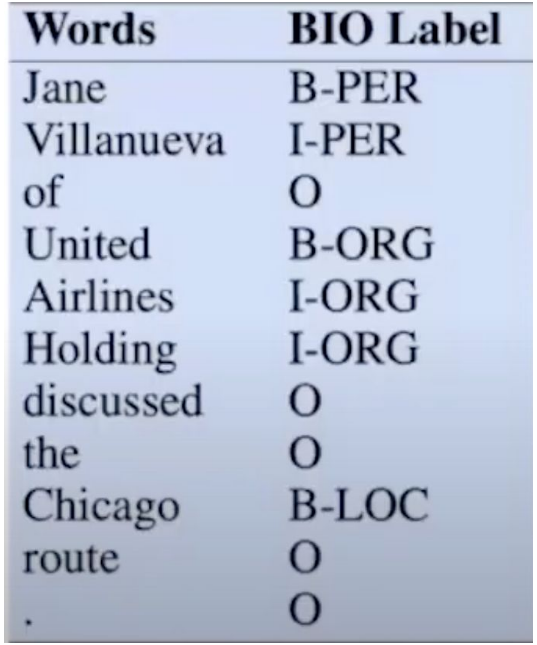
BIO tagging

[PER Jane Villanueva] of [ORG United] , a unit of [ORG United Airlines Holding] , said the fare applies to the [LOC Chicago] route.

B: token that *begins* a span

I: tokens *inside* a span

O: tokens outside of any span



Words	BIO Label
Jane	B-PER
Villanueva	I-PER
of	O
United	B-ORG
Airlines	I-ORG
Holding	I-ORG
discussed	O
the	O
Chicago	B-LOC
route	O
.	O

Text Processing | NER

NER – step 2:

- This step requires the creation of entity categories

Some of the common entity categories are:

- Person
  - Elvis Presley, Cristiano Ronaldo, Barack Obama
- Organization
  - Google, Mastercard, University of Oxford

Text Processing | NER

NER – step 2:

- Time
    - 2022, 18:24, 5pm
- Location
    - Trafalgar Square, MoMA, Machu Picchu
- Work of art
    - Hamlet, Guernica, Exile on Main St., Monalisa

Text Processing | NER

Where is NER Used?

- Human Resources
- Customer Support
- Search and Recommendation Engines
- Content Classification
- Health Care
- Academia

Feature Extraction

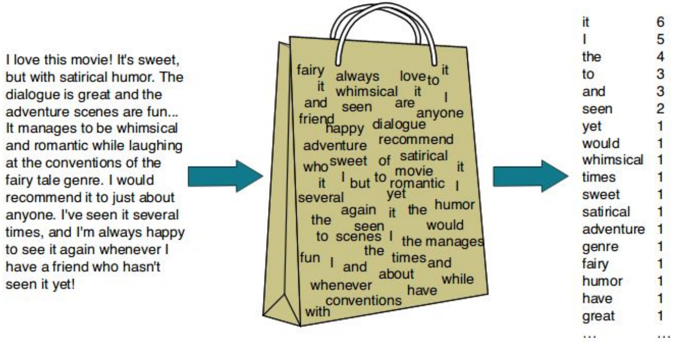
Feature Extraction

- Extract and produce feature representations that are appropriate for the type of NLP task we're trying to accomplish and the type of model we're planning to use

- Examples:
  - Bag of Words
  - TFIDF
  - One-Hot Encoding
  - Word Embedding

Data
Features

Bag of Words



I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

- BoW – Bag of Words
- Bag of words is a Natural Language Processing technique of text modelling
- It is a method of feature extraction with text data
- It is a simple and flexible way of extracting features from documents

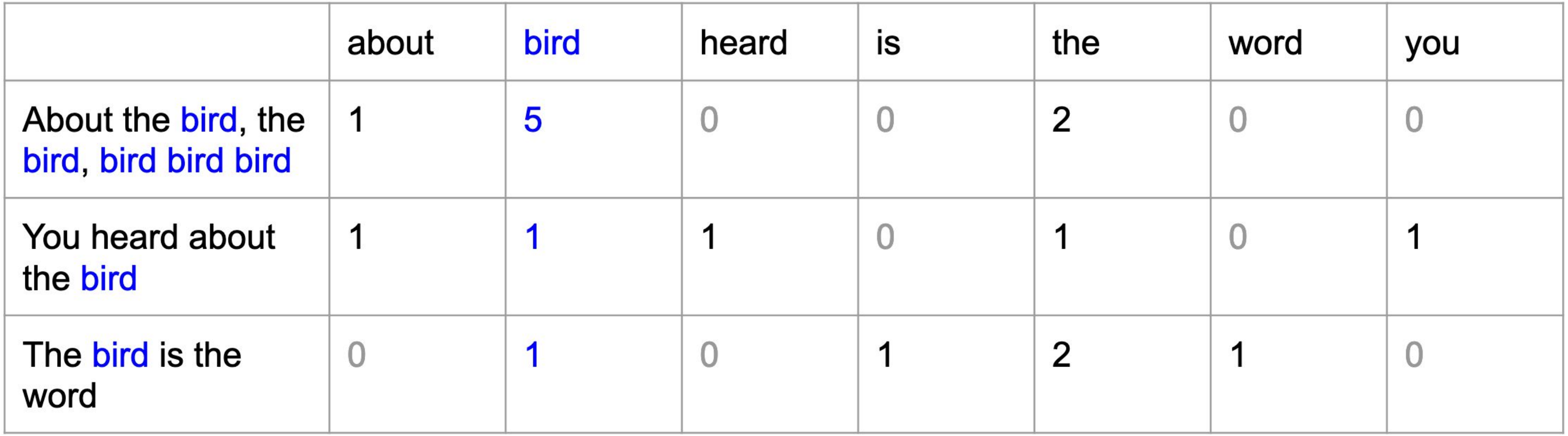
Bag of Words

- A bag of words is a representation of text that describes the occurrence of words within a document

- We just keep track of the word counts and disregard the grammatical details and the word order

- It is called “bag” because any information about the order or structure of the words in the document is disregarded

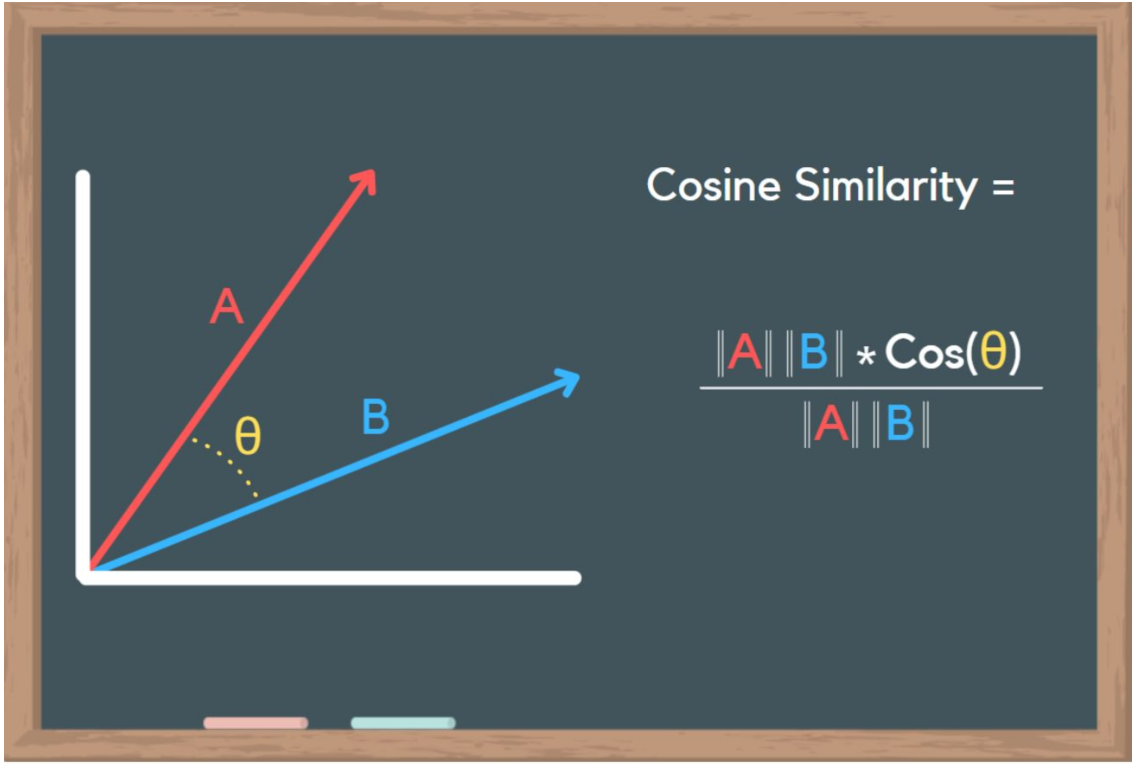
Bag of Words



	about	bird	heard	is	the	word	you
About the bird, the bird, bird bird bird	1	5	0	0	2	0	0
You heard about the bird	1	1	1	0	1	0	1
The bird is the word	0	1	0	1	2	1	0

- Downside of BoW: It treats every word as being equally important

Bag of Words | Similarities



TF-IDF

- **T**erm **F**requency **I**nverse **D**ocument **F**requency
- It is a statistical measure that evaluates how relevant a word is to a document in a collection of documents
- Usage:
  - Most importantly in automated text analysis
  - It is very useful for scoring words in machine learning algorithms for Natural Language Processing (NLP)

TF-IDF

- TF-IDF was invented for document search and information retrieval

- It works by increasing proportionally to the number of times a word appears in a document but is offset by the number of documents that contain the word

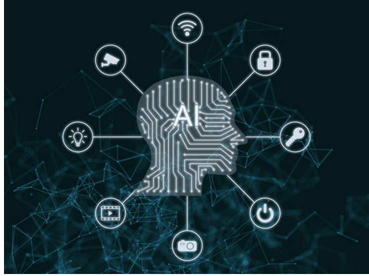
- TF-IDF is achieved by multiplying 2 metrics:
    - How many times a word appears in a document
    - The inverse document frequency of the word across a set of documents

TF-IDF

- TF-IDF will rank the words keeping in mind how frequent they are in all the documents

- Some words could be common in the corpus, like cost in a financial document

- TF-IDF assigns weights to words that signify their relevance in documents



TF-IDF | usage in ML

- The main hurdle that ML with Natural Language faces is that its algorithms usually deal with numbers, and natural language is text.

- It is a fundamental step in the ML process for analyzing data, and choosing the vectorization algorithm that will deliver the results you’re hoping for

- Then, the TF-IDF score can be fed to algorithms such as Naïve Bayes and Support Vector Machines, greatly improving the results of more basic methods like word counts.

TF-IDF | Calculation

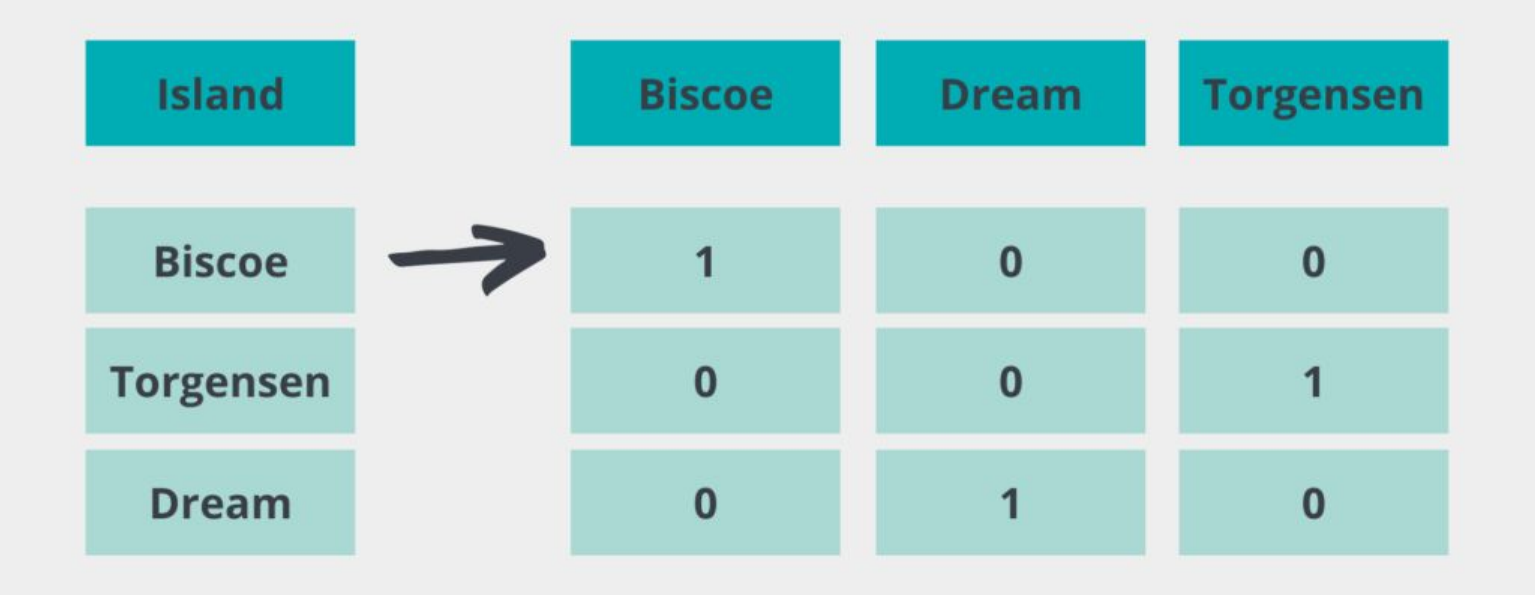
- The **term frequency** of a word in a document:
  - Simplest way to calculate it is a raw count of instances a word appears in a document
  - Other ways are to adjust the frequency, by the length of a document, or by the raw frequency of the most frequent word in a document

- The **inverse document frequency** of the word across a set of documents:
  - This means how common or rare a word is in the entire document set
  - The closer it is to 0, the more common a word is
  - This metric can be calculated by taking the total number of documents and dividing it by the number of documents that contain a word and calculating the logarithm

One Hot Encoding

- Similar to Bag of Words
- The difference is that the columns are the words and the rows each word in the document
- So the row is all zeros except one frequency number at the place of the word
- This works sometimes, but breaks when we have a large vocab

One Hot Encoding



Island
Biscoe
Torgensen
Dream

Biscoe	Dream	Torgensen
1	0	0
0	0	1
0	1	0

Sequence Processing

Challenges of feedforward networks:

- language requires access to information “distant” from the current word
- sliding window makes it hard to learn how “phrases of words” combine together

Solution:

- RNNs (this week)
- transformer networks (next week)

Group Discussion

15 min

Refresher

What is a recurrent neural network?

Give a few examples of an RNN architecture.

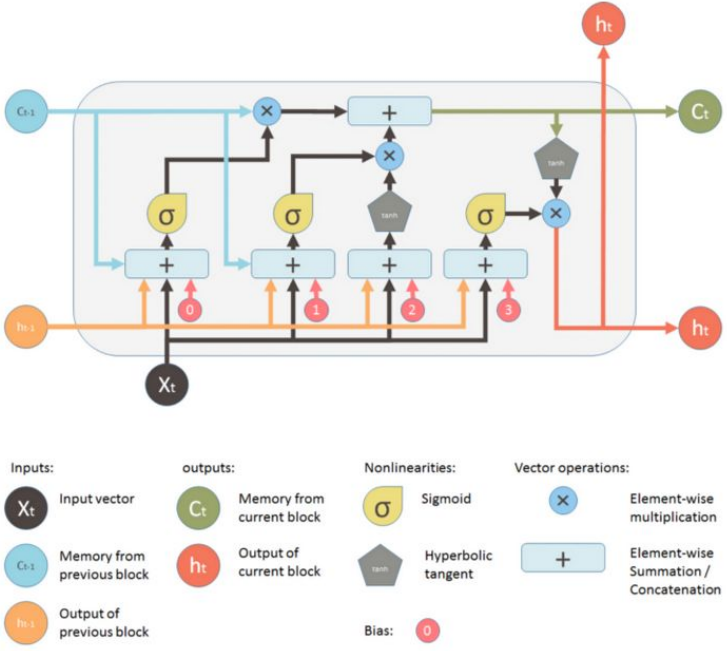
Sequence Processing | LSTM

Bidirectional RNNs

- not only using information from the prior context, but also use words after the time of focus ➜ two RNNs, left-to-right and right-to-left
- concat into a single vector at each point in time

The LSTM network

- adding an explicit context layer to the RNN architecture



Ct-1
ht-1
Xt
ht
Ct
ht
σ
σ
tanh
σ
tanh
0
1
2
3
Inputs:
Xt Input vector
Ct-1 Memory from previous block
ht-1 Output of previous block
outputs:
Ct Memory from current block
ht Output of current block
Nonlinearities:
σ Sigmoid
tanh Hyperbolic tangent
Bias: 0
Vector operations:
× Element-wise multiplication
+ Element-wise Summation / Concatenation

image source

Sequence Processing | LSTM

- Other Applications:
  - sentiment analysis and topic modeling
  - POS tagging
  - text generation

NLP Evaluation Metrics

- Determining whether the model being used for a specific task is successful depends on 2 key factors:
    - If the evaluation metric, we have selected is the correct one for our problem
    - If we are following the correct evaluation process

- Types of evaluation:
    - Intrinsic Evaluation: Focuses on intermediary objectives (i.e. the performance of an NLP component on a defined subtask)
    - Extrinsic Evaluation: Focuses on the performance of the final objective (i.e. the performance of the component on the complete application)

Evaluation Metrics | examples

- Accuracy | Recall | Precision | F1 Score
- Mean Average Precision (MAP)
- Mean Absolute Percentage Error (MAPE)
- Root Mean Squared Error (RMSE)
- Area Under the Curve (AUC)
    - AUC helps quantify the model's ability to separate the classes by capturing the count of positive predictions which are correct against the count of positive predictions that are incorrect at different thresholds

NLP Evaluation Metrics | examples

- Bilingual Evaluation Understudy (BLEU)
  - Evaluates the quality of text that has been translated by a machine from one natural language to another
- METEOR
  - A precision-based metric for the evaluation of machine-translation output
  - It overcomes some of the pitfalls of the BLEU score, such as exact word matching whilst calculating the precision
- ROUGE
  - As opposed to the BLEU score, ROUGE measures the recall
  - It is typically used for evaluating the quality of generated text and in summarization tasks
- Mean Reciprocal Rank (MRR)
  - Evaluates the responses retrieved in correspondence to a query, given their probability of correctness
  - It is typically used in information retrieval tasks

What questions do you have?

Group Discussion

15 min

Try using ChatGPT (https://chat.openai.com/)

Feedback on Lecture and Concepts?