UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Faculty of Engineering, Built Environment and Information Technology

Fakulteit Ingenieurswese, Bou-omgewing en Inligtingtegnologie / Lefapha la Boetšenere, Tikologo ya Kago le Theknolotši ya Tshedimošo

Inputs:
- A Modupe [PhD Candidate]
- A Moodley [MIT Big Data

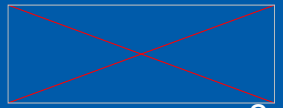
Quick Intro to NLP

Dr. Vukosi Marivate

Make today matter

Data Science for Social Impact

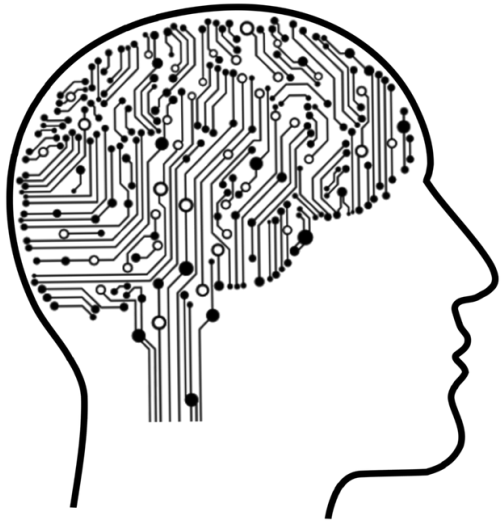
Natural Language Processing



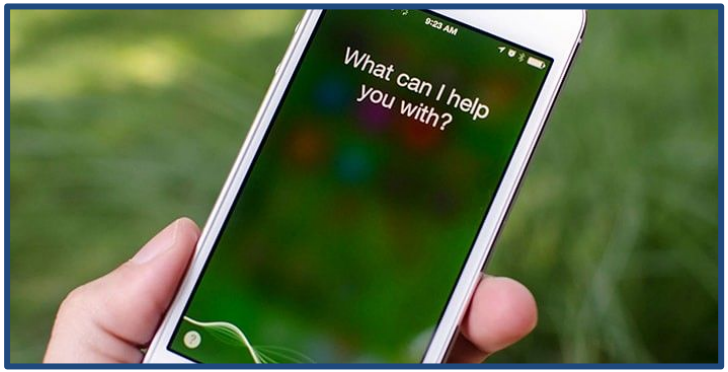
NLP?

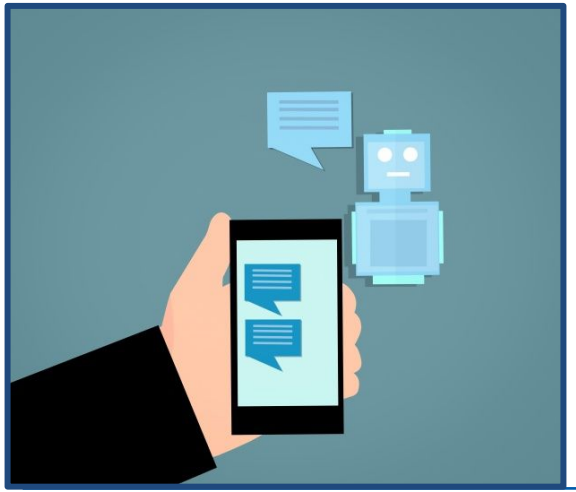
NLP is a broad field, encompassing a variety of tasks, including:

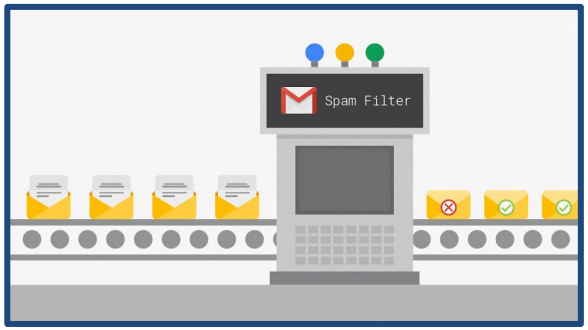
- Part-of-speech tagging: noun, verb, adjective, etc.)
- Named entity recognition NER): person names, organizations, locations, etc.
- Question answering
- Speech recognition
- Text-to-speech and Speech-to-text
- Topic modeling
- Sentiment classification
- Language modeling
- Translation



NLP Breakthroughs

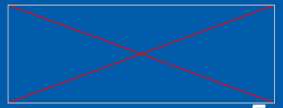






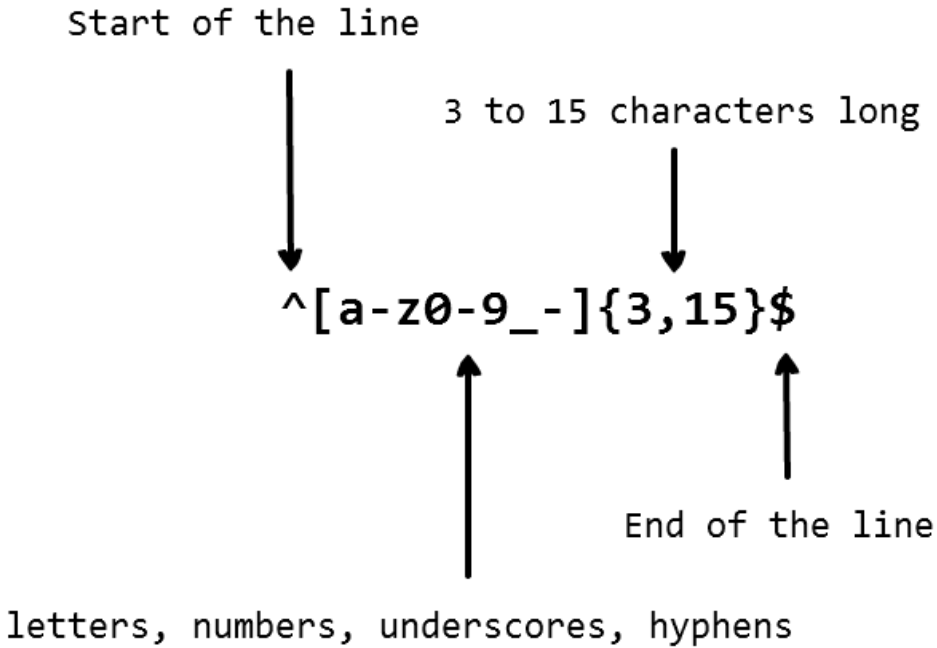
https://gifer.com/en/Ou1t

Basic Text Processing



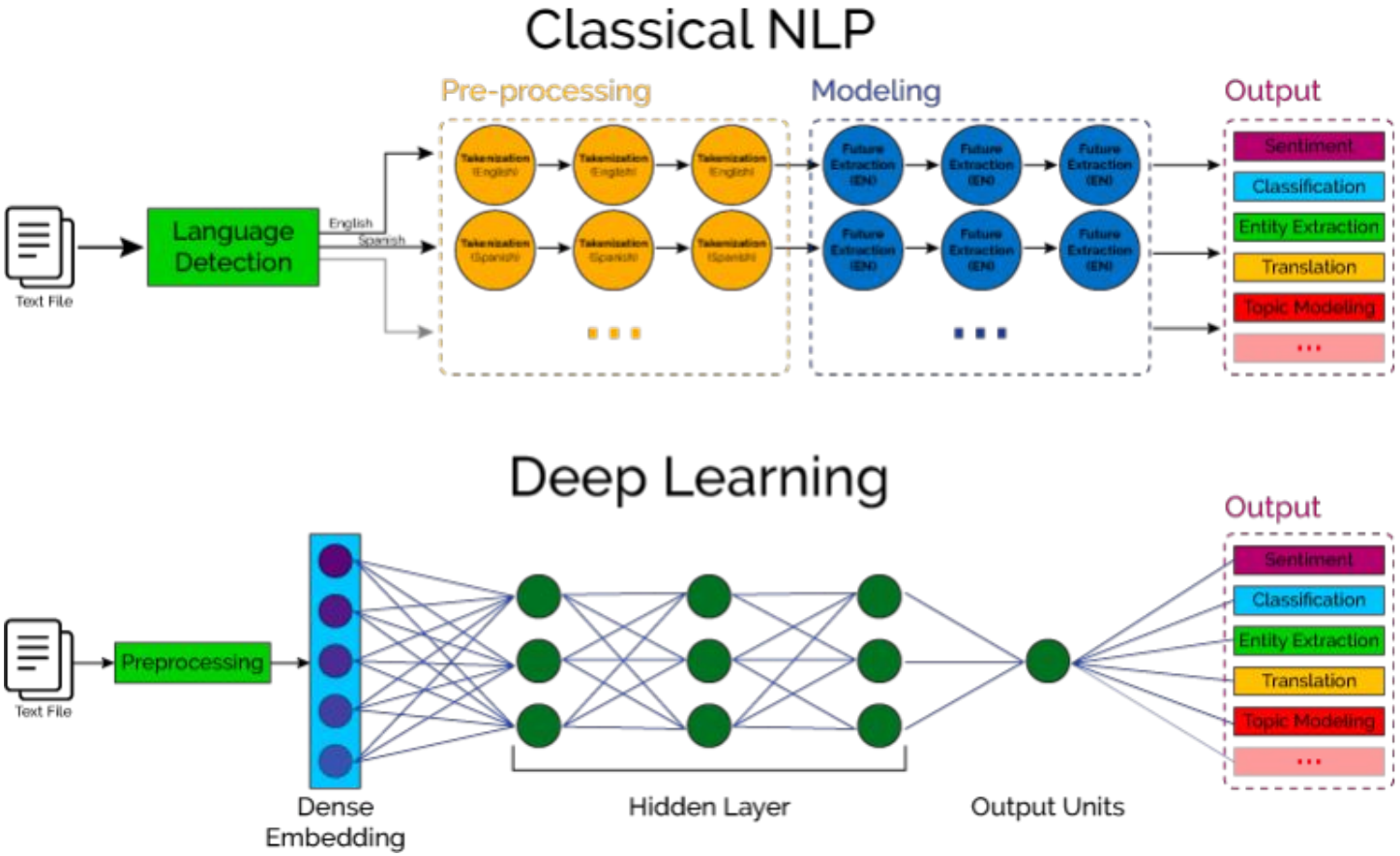
Regular Expressions

Looking for patterns in text. Example below



https://web.stanford.edu/~jurafsky/slp3/
https://www.datacamp.com/courses/regular-expressions-in-python [Chapter 1, also free to do]

Data driven (Statistical) vs. Linguistics

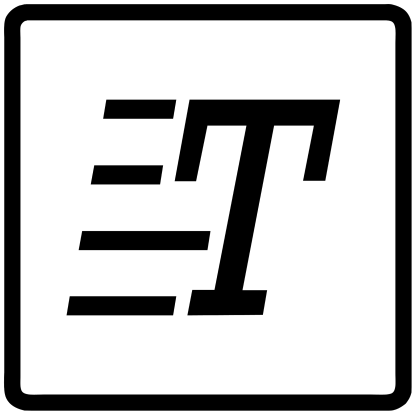


Getting machines to understand: Symbols

We need features!!!

Treat words as **atomic features**

- *car, house, tree, drove, past*



Approach: one hot vector for each symbol

car = [1 0 0 0 0]
house = [0 1 0 0 0]
tree = [0 0 1 0 0]

Sentences?: We can make sentences/combinations

car drove past tree house = [1 1 1 1 1]

Tokenization

Sentence Tokenization

Document

"Wakanda is located in East Africa, although its exact location has varied throughout the nation's publication history: some sources place Wakanda just north of Tanzania, while others – such as Marvel Atlas #2 – show it at the north end of Lake Turkana, in between South Sudan, Uganda, Kenya and Ethiopia (and surrounded by fictional countries like Azania, Canaan, and Narobia). In the Marvel Cinematic Universe (The Black Panther), on-screen maps use the location given in Marvel Atlas #2."

Sentence Tokenization

S1 = Wakanda is located in East Africa, although its exact location has varied throughout the nation's publication history: some sources place Wakanda just north of Tanzania, while others – such as Marvel Atlas #2 – show it at the north end of Lake Turkana, in between South Sudan, Uganda, Kenya and Ethiopia (and surrounded by fictional countries like Azania, Canaan, and Narobia).

S2 = In the Marvel Cinematic Universe (The Black Panther), on-screen maps use the location given in Marvel Atlas #2.

Tokenization

Word Tokenization

Document

"Wakanda is located in East Africa, although its exact location has varied throughout the nation's publication history: some sources place Wakanda just north of Tanzania, while others – such as Marvel Atlas #2 – show it at the north end of Lake Turkana, in between South Sudan, Uganda, Kenya and Ethiopia (and surrounded by fictional countries like Azania, Canaan, and Narobia). In the Marvel Cinematic Universe (The Black Panther), on-screen maps use the location given in Marvel Atlas #2."

Word Tokenization

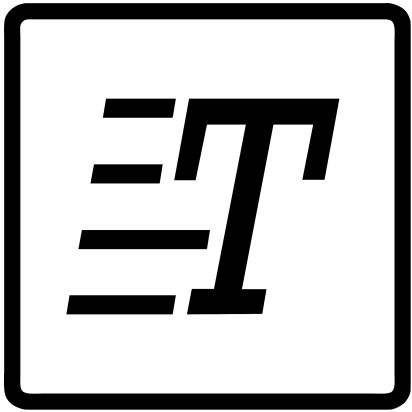
S1 = ['Wakanda', 'is', 'located', 'in', 'East', 'Africa', ',', 'although', 'its', 'exact', 'location', 'has', 'varied', …..]

S2 = ['In', 'the', 'Marvel', 'Cinematic', 'Universe', '(', 'The', 'Black', 'Panther', ')', ',',…..]

We could also do character level extraction!!!!!!

Term Frequency

USE CASE Document Retrieval (Simple search Engine)



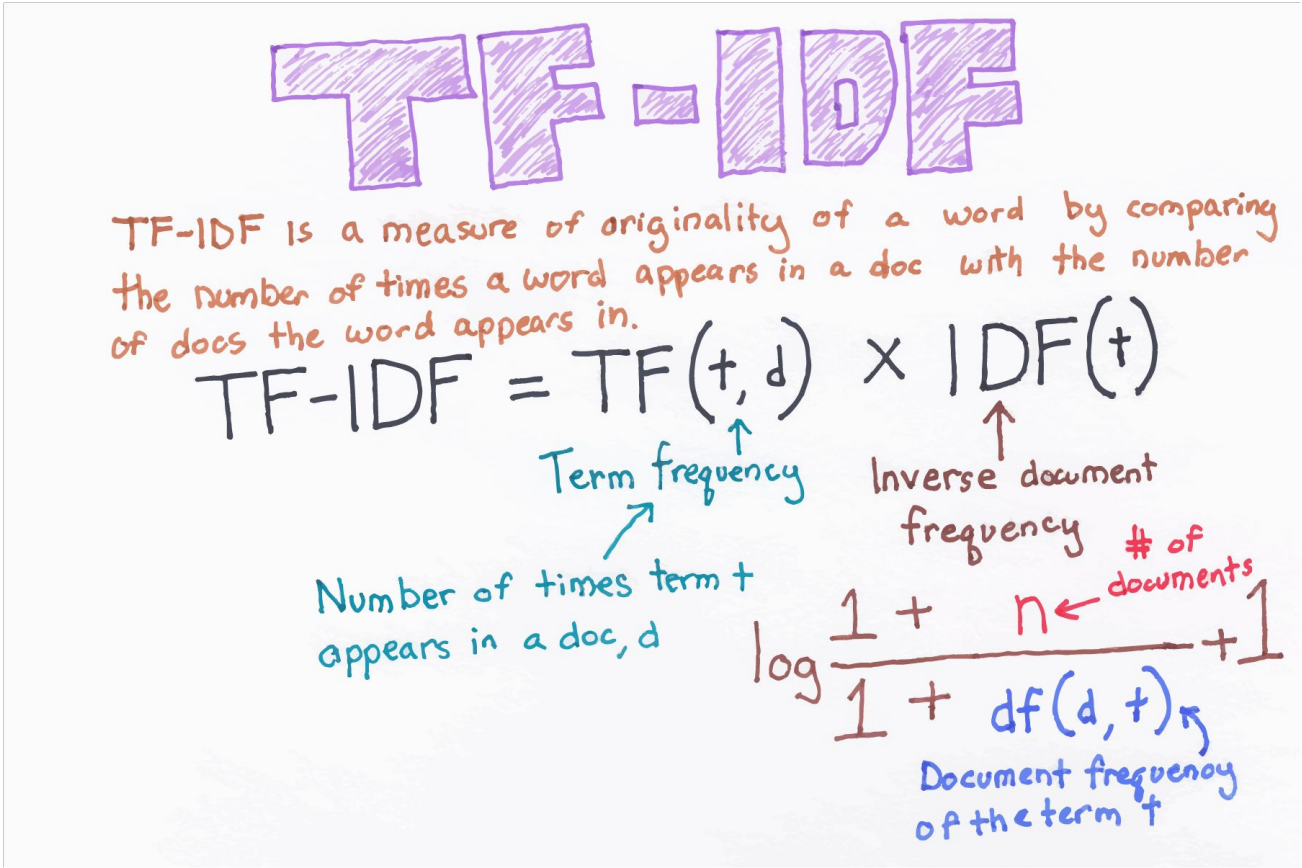
Task:

- Have many documents (**A corpus**)
- We have a search term **Q** (Q made up of symbols)
- Want to return documents relevant to **Q**

Approach

- Count how many times symbols appear in each document.
- Return the documents that have the highest count of the symbols in **Q**

TF-IDF



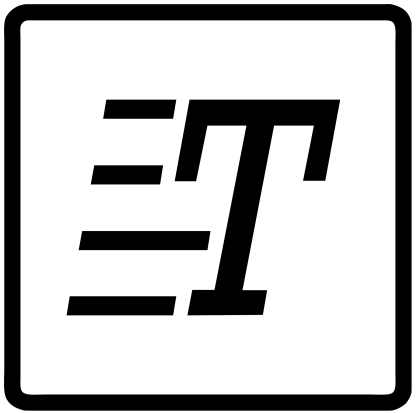
https://chrisalbon.com/machine_learning/preprocessing_text/tf-idf/

Possibilities

Classification

Sentiment Analysis
News categorization
etc.

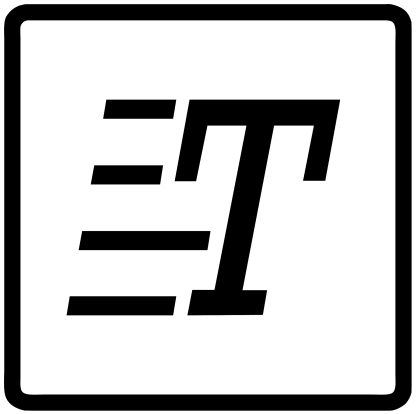
Topic Modelling



Challenges (Some)

Dealing with sequences

Sparse data representation



Processes

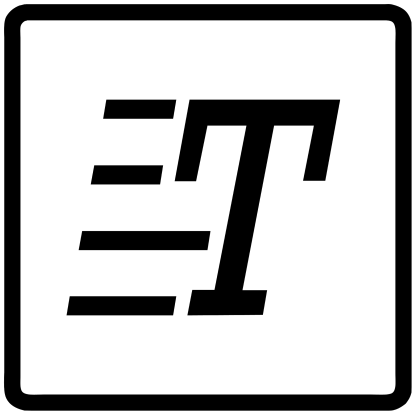
Document -> Symbols [Tokenization]

Symbols -> Vector [Vectorization]

- Frequency
- TF-IDF

Noise?

- Typos
- Misspellings

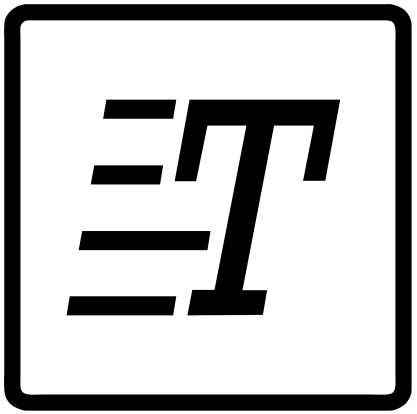


Semantic Meaning

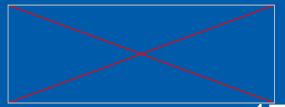
We want terms and phrases that are similar to be treated similarly

Ideas? 💡

- **Synonym generation**
- **Learn a similarity mapping**



ML and NLP



Classification

- Categorising Data
- Spam/Not Spam
- Sentiment Analysis
- Hate Speech

Different choices of feature inputs

Bag-Of-Words/TF-IDF + Logistic Regression/SVM/XGBoost etc.

Word Vectors?

- Different Means
- Recurrent Neural Networks
- Transformers
- Etc.

Sentence Vectors? [Concatenated power means]

https://arxiv.org/abs/1803.01400

Example: News Classification

Bag-Of-Words + Logistic Regression

Input:

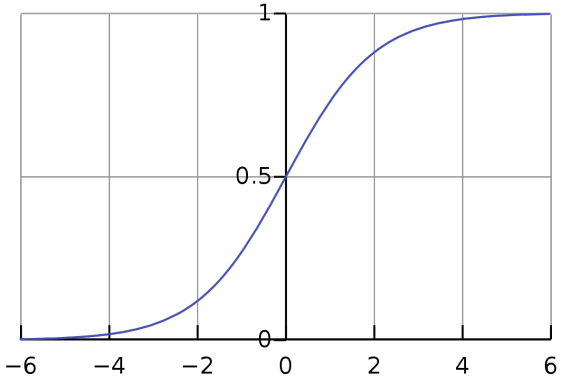
Orlando Pirates were largely on top throughout as they defeated Kaizer Chiefs 2-0 in Saturday's Carling Black Label Cup final at FNB Stadium. Lazarous Kambole started in attack ahead of Leonardo Castro whereas Brilliant Khuzwayo was dropped from the line-up due to injury at the expense of Leonardo Castro. In the early exchanges, Dumisani Zuma threatened Sandilands' goal but his looping shot failed to hit the back of the net.

Example: News Classification

Bag-Of-Words + Logistic Regression

Input:

Orlando Pirates were largely on top throughout as they **defeated Kaizer Chiefs** 2-0 in Saturday's **Carling Black Label Cup** final at **FNB Stadium. Lazarous Kambole** started in attack ahead of **Leonardo Castro** whereas **Brilliant Khuzwayo** was dropped from the line-up due to **injury** at the expense of **Leonardo Castro**. In the early exchanges, **Dumisani Zuma** threatened Sandilands' **goal** but his looping shot failed to hit the **back of the net**.



Output:

Sports - Football [90%]

Sports - Rugby [4%]

Topic Modelling

- Grouping Data
- Clustering
- Extracting Latent Patterns

Different choices of feature inputs

Bag-Of-Words/TF-IDF + Latent Dirichlet Allocation/NMF/LSA

Word Vectors?

- LDA2Vec

Can be used for other downstream processes

LDA

Algorithms:

1. Each document is modeled as a multinomial distribution of topics and each topic is modeled as a multinomial distribution of words

2. LDA assumes that every chunk of text will feed into it contain words that are somehow related. Hence, choosing the right corpus of data is crucial

3. LDA assumes that documents are generated from a mixture of topics an those topics then generate words on their probability distribution

Sample script from python:

```
from sklearn.datasets import fetch_20newsgroups
newsgroups_train = fetch_20newsgroups(subset='train', shuffle = True)
newsgroups_test = fetch_20newsgroups(subset='test', shuffle = True)
```

Topics

gene	0.04
dna	0.02
genetic	0.01
...	

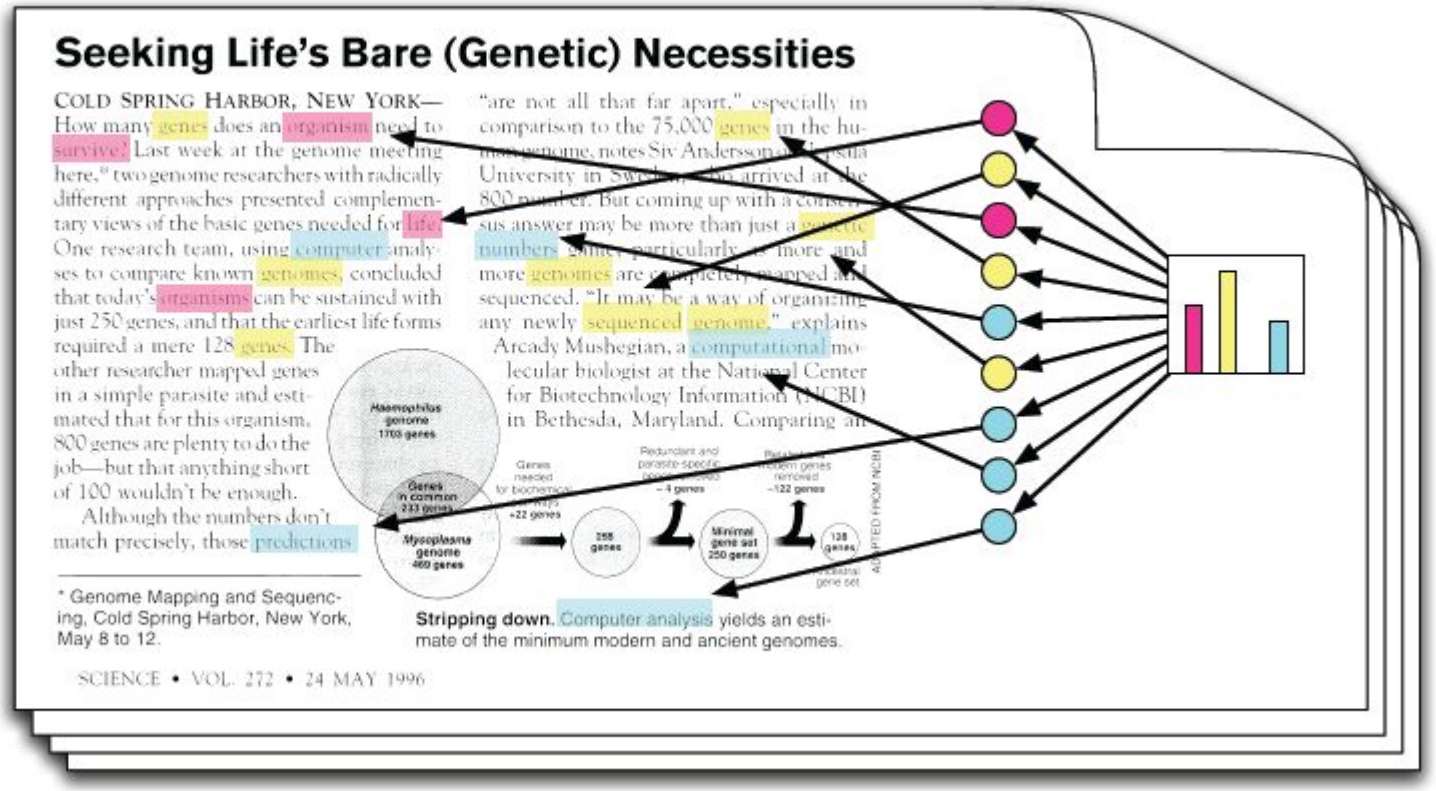
life	0.02
evolve	0.01
organism	0.01
...	

brain	0.04
neuron	0.02
nerve	0.01
...	

data	0.02
number	0.02
computer	0.01
...	

Documents

Topic proportions and assignments



Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

Extracting Topics with LDA

This include the following:

1. **Tokenization:** Split the text into phrases and the sentences into words, convert text to lower letters and remove punctuation\

2. **Stopwords** can be removed, e.g. using Regexp

3. Word are **Lemmatized** --- words in third person can be changed to first person and verbs in past and future tenses are changed into present

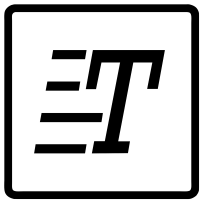
4. Words are **stemmed** --- words are reduced to their root form

NLTK and gensim libraries for preprocessing

```
def lemmatize_stemming(text):
    return stemmer.stem(WordNetLemmatizer().lemmatize(text,  pos='v'))

# Here we tokenize and lemmatize

def preprocess(text):
    result=[]
    for token in gensim.utils.simple_preprocess(text):
        if token not in gensim.parsing.preprocessing.STOPWORDS and
            len(token) > 3:
            result.append(lemmatize_stemming(token))
    return result
```



Results

Original document:

['This', 'disk', 'has', 'failed', 'many', 'times.', 'I', 'would', 'like', 'to', 'get', 'it', 'replaced.']

Tokenized and lemmatized document:

['disk', 'fail', 'time', 'like', 'replac']

Bag of word to convert the input text

Before employing LDA algorithms, or topic modelling. First, tokenization and lemmatization of text using BoW is needed…

E.g., dictionary where the key is the word and value is the number of times that word occurs in the text **Example:**

```
dictionary = gensim.corpora.Dictionary(processed_docs)
```

Then, we can filter the words that occur few times or frequently occur: First construct dictionary object to convert document into BoWs (i.e., for each doc, we create a dictionary to report how many words and frequency):

```
bow_corpus = [dictionary.doc2bow(doc) for doc in processed_docs]
```

NMF: Non-negative Matrix factorization

- Linear-algebra model that factors high-dimensional vectors into a low-dimensionality representation.
- Like Principal component analysis (PCA)
- NMF takes advantage of the fact that the vectors are non-negative
- By factoring them into the lower-dimensional form, NMF forces the coefficients to also be non-negative.

NMF: Algorithm

Given the original matrix **A**, we can obtain two matrices **W** and **H**, such that A= WH.

NMF has an inherent clustering property, such that W and H represent the following information about A:

- A (Document-word matrix) — input that contains which words appear in which documents.
- W (Basis vectors) — the topics (clusters) discovered from the documents.
- H (Coefficient matrix) — the membership weights for the topics in each document.

NMF: Algorithm

We calculate W and H by optimizing over an objective function (like the EM algorithm), updating both W and H iteratively until convergence as follow:

$$\frac{1}{2}||\mathbf{A}-\mathbf{WH}||_{\mathrm{F}}^2=\sum_{i=1}^{n}\sum_{j=1}^{m}(A_{ij}-(WH)_{ij})^2$$

The objective function measure the error of reconstructing between A and the product of its factor W and H using euclidean distance (e.g., you try other distance similarity measure). The rule for W and H is update to obtain:

$$W_{ic}\leftarrow W_{ic}\frac{(\mathbf{A}H)_{ic}}{(\mathbf{WH}H)_{ic}} \qquad H_{cj}\leftarrow H_{cj}\frac{(W\mathbf{A})_{cj}}{(W\mathbf{WH})_{cj}}$$

Language Modelling

- Understanding Language Semantics
- Sequence to Sequence Models

Different choices of feature inputs

Estimate:

$P(w|W) = P(w_{t+1}|w_1, w_2, w_3, w_4)$

t - next word

W - word history

Can be used for:

- Generating text
- Word/Sentence Embeddings

Language Modelling

- Understanding Language Semantics
- Sequence to Sequence Models

Different choices of feature inputs

Estimate:

$P(W) = P(w_1, w_2, w_3, w_4)$

W - word history

Can be used for:

- Generating text
- Word/Sentence Embeddings

Resources

NLP General

- https://github.com/fastai/course-nl
- Stanford Coursera NLP Slides https://web.stanford.edu/~jurafsky/NLPCourseraSlides.html
- Sebastian Ruder Newsletter [http://ruder.io/nlp-news/]
- https://nlpprogress.com

Python Libraries

- SKLearn NLP (Working With Text Data) - URL (Nice tutorial)
- spaCY: Industrial-Strength Natural Language Processing - URL
- NLTK

Thank You

Dr. Vukosi Marivate
vukosi.marivate@cs.up.ac.za
https//dsfsi.github.io
@vukosi