Advanced Machine Learning

Exercise 5 – NLP models

by
Tomer Segal, ID301833833
Millis Sahar, ID300420379

Code Notebooks:

1. Data Extraction
2. Exploratory Analysis
3. Part 1 & 2
4. Part 3
5. Submission file
6. Final Report

Intro:

The world is changing as we speak.
The COVID-19 epidemic has been causing dramatic changes to the way we live our life.
In this exercise we will focus on data related to the COVID-19 and will build unsupervised machine learning models.
The data that you will play with is a big corpus of academic papers, all dealing with COVID-19.
The data is part of a worldwide research challenge, provided by Kaggle and is well described here.

Data Extraction / Preprocessing

The following is the details about the preprocessing stage in chronological order:

1. Load the metadata file and remove articles without pdf files
2. Choose the top 10k most recent articles out of the dataset according to the publish date
3. For each article, extract the following and store in a dataframe:
    - title
    - cord_uid
    - abstract
    - body text
4. Save the dataframe for future usage.

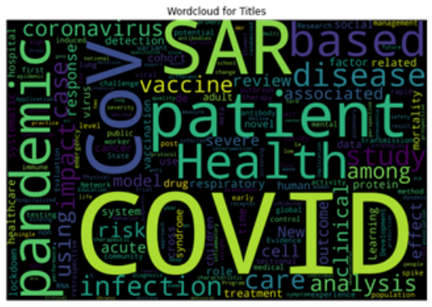
Here is our code.

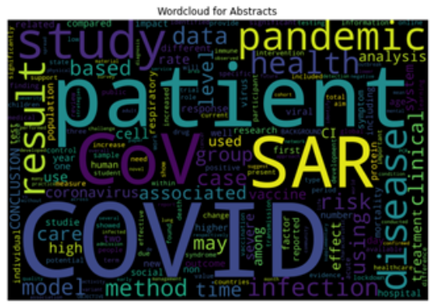
Dataframe sample:

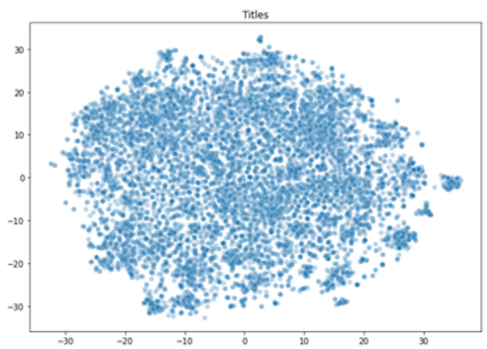
	title	cord_uid	abstract	body_text
0	Chapter 6 Best practices and approaches using ...	rsgdqk1x	This chapter provides ten practical case studi...	Throughout the previous chapters, this book ad...
1	A Comparison: Prediction of Death and Infected...	k58agggx	COVID-19 is a virus causing pneumonia, also kn...	COVID-19 is a virus causing pneumonia, also kn...
2	Environmental implication of personal protecti...	ercubdf9	In the present global health emergency, face m...	Due to the fast development in the domain of c...
3	The first three weeks of lockdown in England: ...	3u446iuy	With the outbreak of COVID-19 being declared a...	The first UK confirmed death from COVID-19 was...
4	Appendix B Climate change and global warming: ...	ora47nlw	Unknown	Climate change and global warming:\nImpacts on...

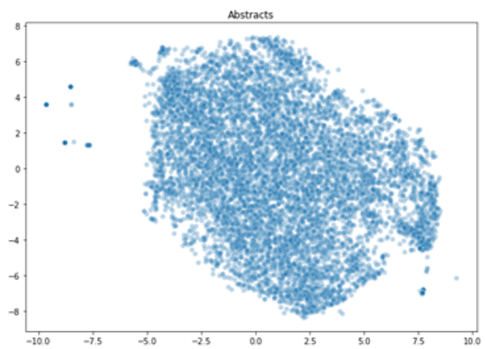
Exploratory Analysis

Here is our code.









Part 1 – comparison model training

The world is changing as we speak.
The COVID-19 epidemic has been causing dramatic changes to the way we live our life.
In this exercise we will focus on data related to the COVID-19 and will build unsupervised machine learning models.

The data that you will play with is a big corpus of academic papers, all dealing with COVID19.
The data is part of a worldwide research challenge, provided by Kaggle and is well described here.
It is the same dataset used for HW3!

Your mission is divided into two parts:

Design and implement **comparison capability between academic papers**.
The comparison capability should be based on natural-language-processing tools and algorithms.

Your comparison model should rely upon a representing schema of each document. You can use the following representation logics (but not limited only to those):

> a. Bag of words
> b. LDA (Topic modeling)
> c. Word embeddings (e.g., Glove, Word2Vec)
> d. Transformers embeddings

Your model need to have the ability to get as input two documents (corona papers in our case) and return a similarity measure between the two.

Indeed a learning phase is required in order to develop such capability, but note that you do not have a tagged corpus.

Here is our code.

Method Description:

We have decided to use `BERT` (Bidirectional Encoder Representations from Transformers) as our compression algorithm.

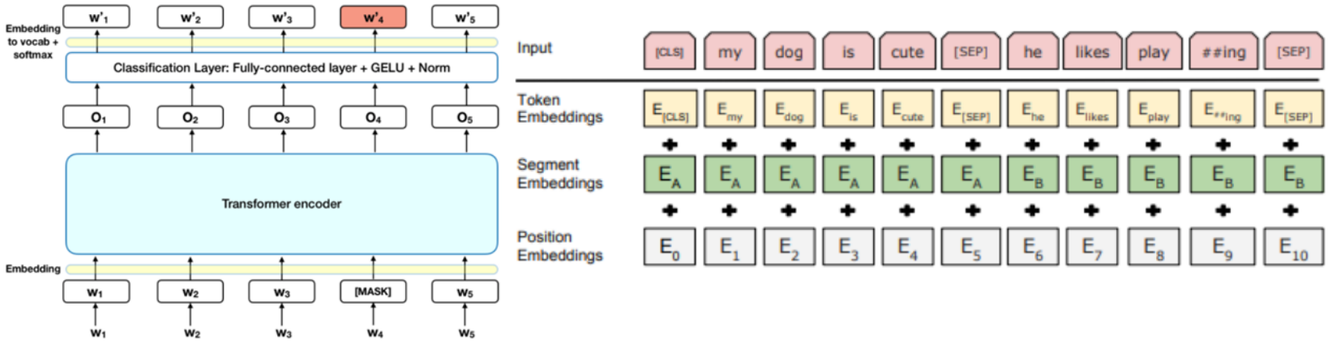
This algorithm produces state of the art results in a wide variety of NLP tasks.
Instead of looking at the text either from left to right or from right to left, BERT's new approach is to look for both directions – Bidirectional.

BERT's paper showed that a language model which is trained bidirectionally can have a deeper sense of language context than single-direction language models.

A small taste of how BERT works – It makes the use of Transformer, a novel architecture that aims to solve sequence-to-sequence tasks while handling long-range dependencies.

The transformer encodes a given sequence of tokens (characters, words, sentences etc.) into a vector called `Embeddings` .
The idea is that similar sequences of tokens will have similar embeddings.

The following is an Illustration of BERT:



Article similarity

In this work, we have used a pre-trained BERT model.
We encoded different data attributions from the given articles dataset (title and abstract) to a 768 vector (embeddings) and used these embeddings to find related articles to the different queries in task id 583.

The following are the different queries in task id 583:

- What has been published about information sharing and inter-sectoral collaboration?
- What has been published about data standards and nomenclature?
- What has been published about governmental public health?
- What do we know about risk communication?

- What has been published about communicating with high-risk populations?
- What has been published to clarify community measures?
- What has been published about equity considerations and problems of inequity?

We have tried two approaches to find the top 10 related articles for each query:

1. `Encode the titles` in the dataset using the pre-trained BERT and find the top 10 related articles by using cosine similarity between each encoded query to each encoded title.
2. `Encode the abstracts` in the dataset using the pre-trained BERT and find the top 10 related articles by using cosine similarity between each encoded query to each encoded abstract.

Part 2 – comparison model analysis and evaluation

Here is our code.

Article Clustering

We then tried to try and cluster the embeddings of the two approaches using K-Means.

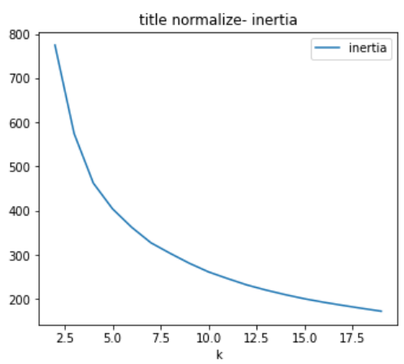
For each approach we performed the following:

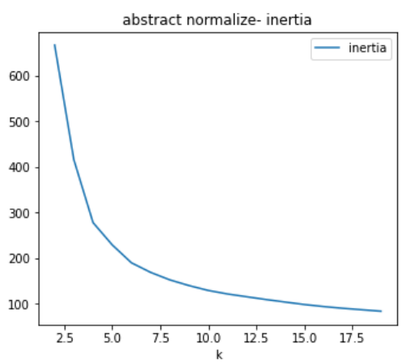
1. Perform dimensionality reduction using PCA to reduce the embeddings dimensions from 768 to 3 dimensions.
2. Perform K-Means algorithm on different number of clusters.
3. Plot the `Inertia` for each configuration of number of clusters.

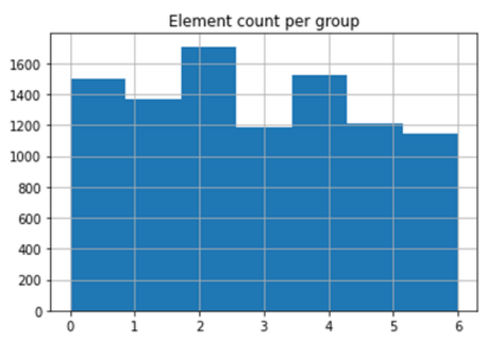
Although is gave us inaccurate results, We also tried to use the `Silhouette coefficient`. Which is a measure of cluster cohesion and separation. And we choose to ignore it, and in our final version, did not used it.

We have used the implementation of inertia from the scikit-learn library, which gives the inerthia for the fitted KMeans. A lower inertia implies for a better clustering.

The following is the result of the clustering:







Using the Elbow method - We can see that the best Ks are in the range of [5-9].
*Due to the size of the dataset we choose **k=7**.*

We also run a TF-IDF to get popuular words, and filter out Stopwods & very common words (e.g COVID,2020,etc.).

For exmaple:

```
group #0 :      ['care', 'clinical', 'health', 'impact', 'patients', 'review', 'study', 'using']
group #1 :      ['associated', 'disease', 'health', 'impact', 'infection', 'lockdown', 'mortality', 'patients', 'risk', 'study']
group #2 :      ['care', 'clinical', 'disease', 'health', 'hospital', 'impact', 'infection', 'patient', 'patients', 'study']
group #3 :      ['approach', 'data', 'future', 'learning', 'model', 'social', 'study', 'using']
group #4 :      ['coronavirus', 'disease', 'human', 'infection', 'patients', 'study', 'vaccine', 'viral', 'virus']
group #5 :      ['antibody', 'cell', 'detection', 'human', 'novel', 'protein', 'rna', 'spike', 'using']
group #6 :      ['global', 'impact', 'model', 'new', 'risk', 'role', 'social', 'study']
```

After trying to understand the clusters using sampling and our own knowledge, we can asume these are the clusters:

- Group #0 - Medical
- Group #1 - Research & study
- Group #2 - Hospitals
- Group #3 - Social aspect
- Group #4 - Diseases
- Group #5 - Very Medical
- Group #6 - Social aspect

Model Evaluation

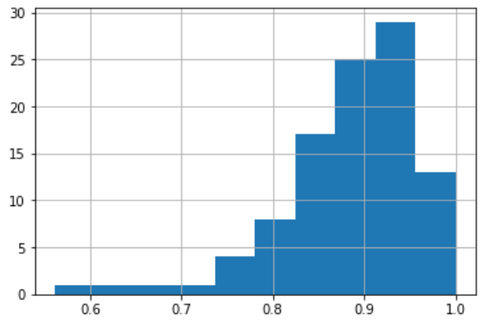
We will use Augmentation! Our augmentation will be based on Gensim's Word2Vec. We'll also consider an augmentation based on translation.

For example:

```
original title >>>  Synthetic protein antigens for COVID-19 diagnostics
augment title  >>>  synthetic protein epitope forthe covid-19 diagnostic_assays
titles similarity:  0.93439543
```

And the similarity Score for all the dataset:

mean score: 0.8857441383600235



GREAT! While using augmentation on the titles, and then perform our similarity - we get very good results!

(We also tried augmentation using 'translation')

An additinal ay will be to check the similarity "by hand":

```
original title:  Inertial Microfluidics Enabling Clinical Research
```

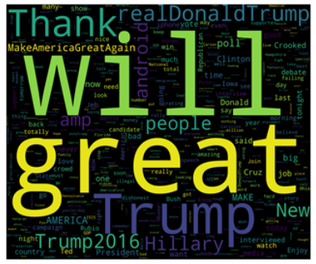
	score	title
0	1.000000	Inertial Microfluidics Enabling Clinical Research
1	0.785904	Myocardial Work: Methodology and Clinical Applications
2	0.768198	Roles of Microglial Ion Channel in Neurodegenerative Diseases
3	0.767849	A Data-driven Framework for Learning and Visualizing Characteristics of Thrombotic Event Phenotypes from Clinical Texts
4	0.765363	Application of nanotechnology in drug delivery systems for respiratory diseases
5	0.757984	Embedding clinical trials within routine health-care delivery: Challenges and opportunities
6	0.753742	Evaluating Dissemination and Implementation Strategies to Develop Clinical Software
7	0.753719	Short-Pulse Lasers: A Versatile Tool in Creating Novel Nano-/Micro-Structures and Compositional Analysis for Healthcare and Wellbeing Challenges
8	0.752917	Telehealth in Multiple Sclerosis Clinical Care and Research
9	0.751083	Optimization of the ASPIRE Spherical Parallel Rehabilitation Robot Based on Its Clinical Evaluation

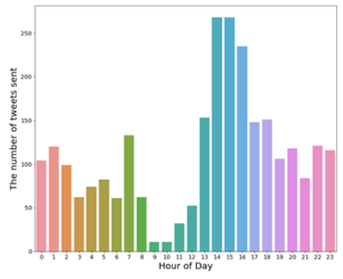
As we can see, the original title get a perfect score - due to it being similar to itself.

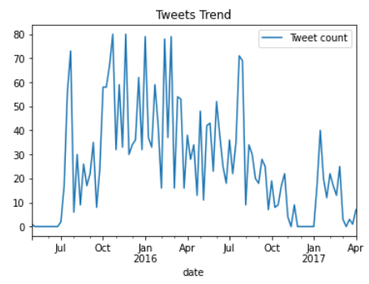
The closest 9 other titles looks kina similar.
Although, this is a domain expert job, we can "feel" the similarity between these titles.

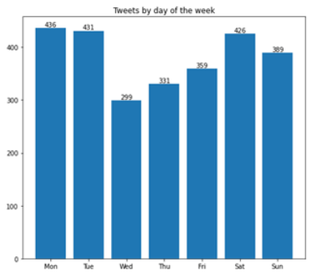
Part 3 – classification algorithm

We did a lot of `Feature Engineering` on the dataset.









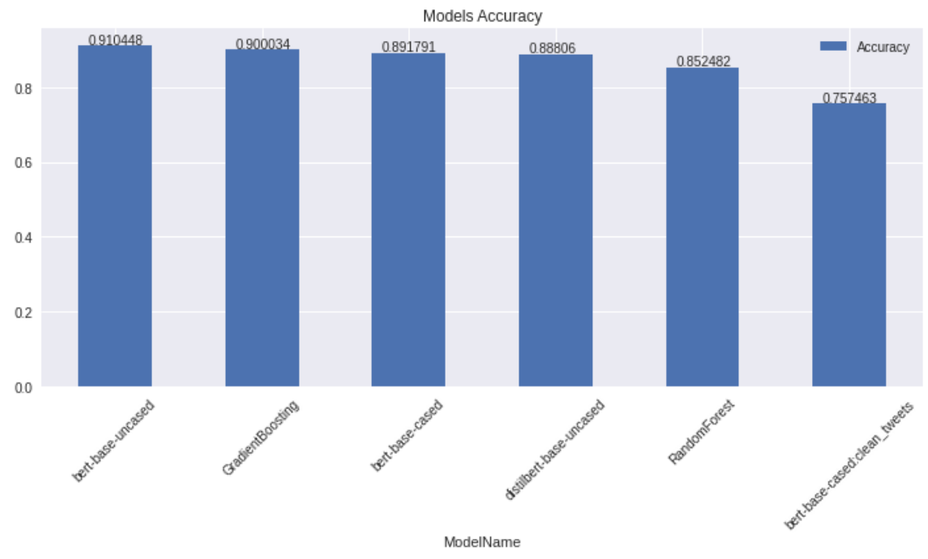
And got a nice ML baseline:

- RF
  - mean score: 0.8524815062528616
  - max score: 0.8712574850299402
- GB
  - mean score: 0.9000338902405085
  - max score: 0.9176646706586826

Then, we used **NLP techniques** and **fine-tuned different language models**.

Please reference our detailed explanation on `Embeddings & BERT models` in the first part of the report.

And here are the results:



Our Chosen model is - `bert-base-uncased`

and we used it to get the predictions for the teset set.

- Count for `TR` is 592
- Count for `NTR` is 274

And you can find the submission file here.

	ID	Tweet	prediction
0	768083669550366720	It is being reported by virtually everyone, an...	TR
1	768097204376510464	Hillary Clinton strongly stated that there wa...	TR
2	768119463421943808	President Obama should have gone to Louisiana ...	TR
3	768125054584393729	Join me in Tampa, Florida- tomorrow at 1pmE! T...	NTR
4	768196613680398336	In Austin, Texas with some of our amazing Bord...	NTR

References:

- Doc2Vec
- Semantic Search
- BERT Language Model
- Semantic Corpus Search
- Taboola - Article Similarity
- BERT Sentence Embeddings
- BERT,MeSH,Knowledge Graph

Kaggle's Inspirations:

- Trump's Tweet Statistics
- Trump's Tweet Visualization
- Feature Engineering for NLP

HuggingFace's References:

- HugginFace's Trainer
- HuggingFace's Pretrained Models
- Finetuning HuggingFace's Transformers

Thanks for reading :)