

**Text Analytics in Sustainable Development Goals (SDGs)**

AUEB M.Sc. in Business Analytics (part-time)

Machine Learning and Content Analytics

Instructor: Haris Papageorgiou

Team Members:

Lianos Alexandros – p2821909

Marinos Eustratios – p2821913

Papanikou Lydia – p2821916

Table of Contents

[Project Description 3](#_Toc53957137)

[Mission 3](#_Toc53957138)

[Data 4](#_Toc53957139)

[Label Encoder 6](#_Toc53957140)

[Basic Preprocessing of text data 6](#_Toc53957141)

[Tokenizing text data using Word Embeddings 6](#_Toc53957142)

[Keras Tokenizer 6](#_Toc53957143)

[Splitting/Shuffling Dataset Procedure 6](#_Toc53957144)

[Methodology 7](#_Toc53957145)

[Multi-Layer Perceptron Neural Network 7](#_Toc53957146)

[Convolutional Neural Network Architecture 7](#_Toc53957147)

[Regarding CNNs' Layers 7](#_Toc53957148)

[Variations - Hyperparameter Tuning 7](#_Toc53957149)

[Keras Embedding Layer 7](#_Toc53957150)

[Dropout Regularization 8](#_Toc53957151)

[Batch Normalization 8](#_Toc53957152)

[Activation Functions 8](#_Toc53957153)

[Loss functions 8](#_Toc53957154)

[Keras Callbacks 9](#_Toc53957155)

[Configuration of the Model 9](#_Toc53957156)

[Results 11](#_Toc53957157)

[Members/Roles 12](#_Toc53957158)

[Bibliography 13](#_Toc53957159)

[Time Plan 13](#_Toc53957160)

[Contact Persons 13](#_Toc53957161)

[Comments/Problems 14](#_Toc53957162)

# Project Description

The challenge of this project is to create a support tool for the academic researcher/scientist who can contribute to making better use of existing scientific literature/research to develop his/her work. The main idea is to train a Neural Network with pre-classified data and create an algorithmic model so that, by entering new data, we can achieve the greatest predictability in new and future publications.

Such tools are already present in the business world and are widely used in various industries. A characteristic example is the biomedical science, where you can get notified of knowledge you need to augment your research.

However, there is a growing interest among governmental researchers and academic institutions in tracking the performance of Sustainable Development Goals (SDGs). To help this scientific research, we can use machine learning algorithms to analyze and connect millions of papers, to measure how SDGs are connected to different research themes.

This tool could be part of the United Nations website, where the user can navigate in a user-friendly platform and focus on specific target indicators.

# Mission

The United Nations is a source of big data in the form of text. Between the many resolutions, speeches, meetings, conferences, studies, reports, and internal regulations that exist and that are produced each year, the UN is awash in text.

However, very few people are in a position to see much more than a small sliver of specialized text. Even fewer can parse the various streams into a coherent and useful picture. What is needed is a quick and objective way to analyse large quantities of United Nations publications according to a desired criteria.

So, our scope is to provide a solution by introducing a proof-of-concept classification system that measures the alignment of publications with each of the pre-defined research themes.

Using machine learning algorithms to analyse digital texts has many advantages. Algorithms can be used at scale with objectivity and can help identify patterns across publications and over time. This approach can also serve as a tool to explore and discover new texts, and to inform the direction of future research. More importantly, this method hopefully inspires other efforts to use modern data analytics to better understand the body of work of the United Nations.

To build such an algorithm however, the main issue that must be tackled is the proper annotation of the papers examined in order to determine the research theme of them.

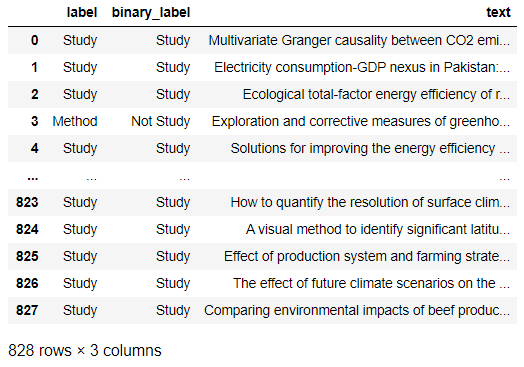
# Data

After the annotation process was completed, the raw data consisted of papers referring to several SDGs and had the following format:

- a folder containing all the papers in separate csv files

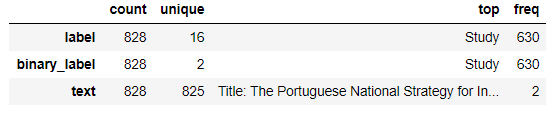
- a csv file with the correspondence of the class (research theme) to every paper

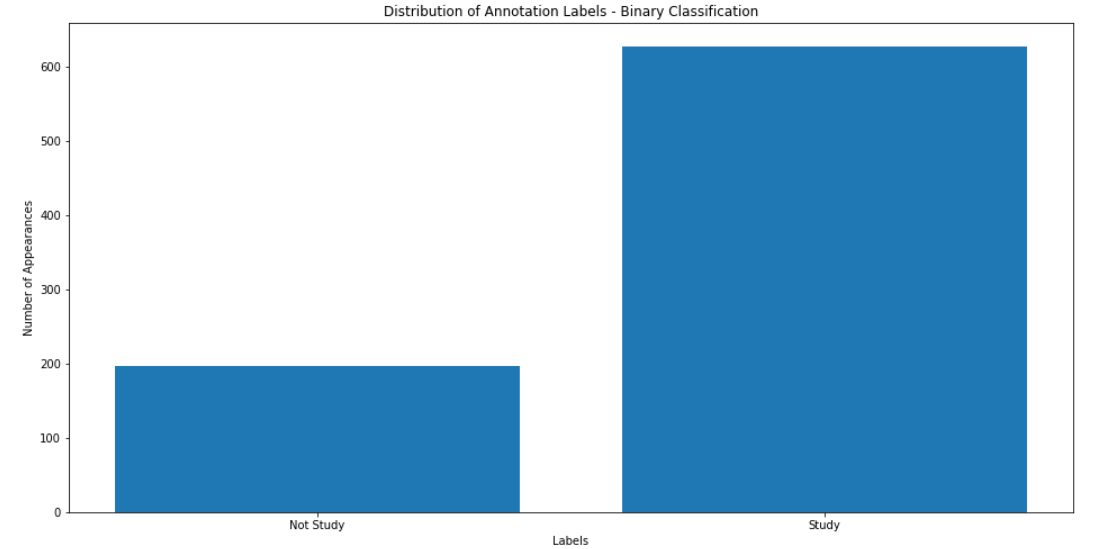
After running a parser to these files, and making the necessary transformations described in the 'parser\_eda\_extract.ipynb' file, we extract a dataframe consisting of 828 papers (rows) and 3 columns as shown below.

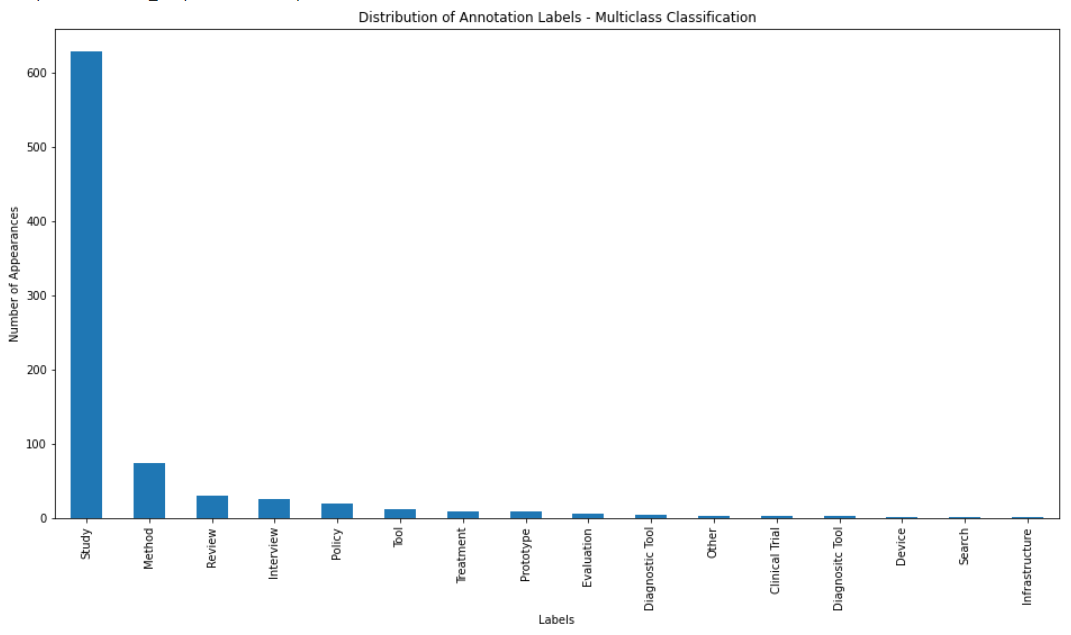


The next step is to perform Exploratory Analysis in our data. We make sure that there are no missing values and remove some duplicate papers during the process. Our final dataset now consists of 825 unique papers. Finally, we create two separate dataframes for binary and multi-class classification purposes, before extracting them to different csv files.

The tables and figures below help us get an overview of our data and reach useful conclusions.







In the case of Binary classification, data are divided into 2 categories (study, no study). On the other hand, in the case of Multiclass classification of 15 categories, the 9 of them were selected as statistically significant, while the others were rejected. In both approaches, we encountered the problem of imbalanced classes. Imbalanced data typically refers to a problem with classification problems where the classes are not represented equally.

Most classification datasets do not have exactly equal number of instances in each class, but a small difference often does not matter. It is the case where your accuracy measures tell the story that you have excellent accuracy (such as 90%), but the accuracy is only reflecting the underlying class distribution. It is very common, because classification accuracy is often the first measure we use when evaluating models on our classification problems.

## Label Encoder

We have to convert the categorical text output data into model-understandable numerical data. *Label Encoder* is part of the SciKit Learn library and it is used to convert categorical data, or text data, into numbers, which our predictive models can better understand.

## Basic Preprocessing of text data

The papers that comprise our dataset, originate from an annotation tool. This means that most of heavy preprocessing, such as special characters, tags, html\code\xml etc., was conducted during the annotation process. Even so, we continue with the some basic preprocess methods, so as to better modify our dataset for the text classification that takes place later.

## Tokenizing text data using Word Embeddings

We cannot feed raw text directly into deep learning models. Text data must be encoded as numbers to be used as input or output for machine learning and deep learning models. Words are called tokens and the process of splitting text into tokens is called tokenization.

A word embedding is a class of approaches for representing words and documents using a dense vector representation. It is an improvement over more the traditional bag-of-word model encoding schemes where large sparse vectors were used to represent each word or to score each word within a vector to represent an entire vocabulary. These representations were sparse because the vocabularies were vast and a given word or document would be represented by a large vector comprised mostly of zero values.

Instead, in an embedding, words are represented by dense vectors where a vector represents the projection of the word into a continuous vector space. The position of a word within the vector space is learned from text and is based on the words that surround the word when it is used. The position of a word in the learned vector space is referred to as its embedding.

For this deep learning project, we will use 2 methods of learning word embeddings:

* GloVe (Using Pre-Trained GloVe Embedding)
* Self-learned as part of a deep learning model (Learning an Embedding)

## Keras Tokenizer

To be added VERY SOON

## Splitting/Shuffling Dataset Procedure

To be added VERY SOON

# Methodology

## Multi-Layer Perceptron Neural Network

Multi-Layer Perceptrons, or MLPs for short, are the classical type of neural network.

They are comprised of one or more layers of neurons. Data is fed to the input layer, there may be one or more hidden layers providing levels of abstraction, and predictions are made on the output layer, also called the visible layer.

## Convolutional Neural Network Architecture

Convolutional Neural Networks, or CNNs, were designed to map image data to an output variable. They have revolutionized image classification and computer vision by being able to extract features from images and using them in neural networks.

The benefit of using CNNs is their ability to develop an internal representation of a two-dimensional image. This allows the model to learn position and scale in variant structures in the data, which is important when working with images.

More generally, CNNs work well with data that has a spatial relationship. The CNN input is traditionally two-dimensional, a field or matrix, but can also be changed to be one-dimensional, allowing it to develop an internal representation of a one-dimensional sequence.

This allows the CNN to be used more generally on other types of data that has a spatial relationship. For example, there is an order relationship between words in a document of text.

### Regarding CNNs' Layers

A CNN has hidden layers which are called convolutional layers. When you think of images, a computer has to deal with a two-dimensional matrix of numbers and therefore you need some way to detect features in this matrix. These convolutional layers are able to detect edges, corners and other kinds of textures which makes them such a special tool. The convolutional layer consists of multiple filters which are slid across the image and are able to detect specific features.

When you are working with sequential data, like text, you work with one dimensional convolutions, but the idea and the application stays the same. You still want to pick up on patterns in the sequence which become more complex with each added convolutional layer.

## Variations - Hyperparameter Tuning

### Keras Embedding Layer

Keras offers an Embedding layer that can be used for neural networks on text data with either approaches:

* Learning an Embedding
* Using Pre-Trained GloVe Embedding

It requires that the input data be integer encoded, so that each word is represented by a unique integer. This data preparation step was performed using the Tokenizer API also provided with Keras.

The Embedding layer is initialized with random weights and will learn an embedding for all of the words in the training dataset.

The Embedding layer is defined as the first hidden layer of a network. It must specify 3 arguments:

* **input\_dim:** This is the size of the vocabulary in the text data. For example, if your data is integer encoded to values between 0-10, then the size of the vocabulary would be 11 words.
* **output\_dim:** This is the size of the vector space in which words will be embedded. It defines the size of the output vectors from this layer for each word. For example, it could be 32 or 100 or even larger.
* **input\_length:** This is the length of input sequences, as you would define for any input layer of a Keras model. For example, if all of your input documents are comprised of 1000 words, this would be 1000.

### Dropout Regularization

Dropout Regularization is a computationally cheap way to regularize a deep neural network.

Dropout works by probabilistically removing, or “dropping out,” inputs to a layer, which may be input variables in the data sample or activations from a previous layer. It has the effect of simulating a large number of networks with very different network structure and, in turn, making nodes in the network generally more robust to the inputs.

Dropout value needs finetuning because a probability too low has minimal effect and a value too high results in under-learning by the network. Dropout can be used on visible as well as hidden units. Application of dropout at each layer of the network has shown good results.

### Batch Normalization

Batch Normalization is just another layer, so you can use it as such to create your desired network architecture. It is a technique designed to automatically standardize the inputs to a layer in a deep learning neural network.

The general use case is to use BN between the linear and non-linear layers in your network, because it normalizes the input to your activation function, so that you're centered in the linear section of the activation function (such as Sigmoid) . It applies a transformation that maintains the mean activation close to 0 and the activation standard deviation close to 1.

Once implemented, batch normalization has the effect of dramatically accelerating the training process of a neural network, and in some cases improves the performance of the model via a modest regularization effect.

### Activation Functions

Activation functions, in layman's terms, are a choice that you must make for each layer:

* **relu:** Very popular neural network activation function. Used for hidden layers, cannot output negative values. No trainable parameters
* **sigmoid:** Classic neural network activation. Often used on output layer of a binary classifier.

### Loss functions

#### BinaryCrossentropy

Computes the cross-entropy loss between true labels and predicted labels.

Use this cross-entropy loss when there are only two label classes (assumed to be 0 and 1). For each example, there should be a single floating-point value per prediction.

### Keras Callbacks

#### Model Checkpoint

Save the current weights of the model at different points during training.

#### Early Stopping

Interrupt training once a target metric being monitored has stopped improving for a fixed number of epochs. For instance, this callback allows you to interrupt training as soon as you start overfitting, thus avoiding having to retrain your model for a smaller number of epochs.

#### ReduceLROnPlateau

Reduce learning rate when a metric has stopped improving.

### Configuration of the Model

The model that we will create is a sequential model meaning that each layer that we add per line will use as input the output of the former layer added to the model. We have to specify:

* **embedding\_dim:** how large we want the word vectors to be. A 50 dimensional vector is able to capture good embeddings even for quite large vocabularies
* **max\_words:** for how many words we want embeddings
* **maxlen:** how long our sequences are

#### Training Parameter

Regarding **batch size** and **epochs**, both are integer values:

The batch size is a hyperparameter that controls the number of training samples to work through before the model’s internal parameters are updated.

The number of epochs is a hyperparameter that controls the number of complete passes through the training dataset

#### Neurons and Layers

Usually it is not a good idea to use less neurons on intermediate layers. Small layers may act as information bottlenecks, permanently dropping relevant information.

# Results

To be added VERY SOON

# Members/Roles

Our team consists of three (3) members with programming and business background, so the roles are well-defined. Efstratios Marinos and Alexandros Lianos operate this project as Data/Machine Learning Engineers and Lydia Papanikou as Business Analyst.

The members of the team dealt with all aspects of the project, but due to limited time restrictions and in the context of time management, each member focused on the responsibilities below:

* Alexandros Lianos: pre-processing of the dataset
* Efstratios Marinos: variations, hyperparameter tuning
* Lydia Papanikou: business implementation and presentation

A brief background of the team:

**Efstratios Marinos**

Graduate of Digital Systems | Communication Systems and Networks at University of Piraeus

DevOps Engineer in Piraeus Bank | Winbank

* Responsible of the release management lifecycle of web applications
* Investigating systems sustainably through mechanisms like automation, so as to avoid performing repetitive tasks.
* Measuring and monitoring progress to ensure application releases are delivered on time and within budget, and that they meet or exceed expectations.

**Alexandros Lianos**

Graduate of Business Administrator at University of Piraeus

Media Planner in AB Vassilopoulos | Ahold Delhaize Group

* Interacting with internal departments to understand their needs and wants before formulating a media strategy
* Recommending media plans to ensure marketing campaigns reach their target audience in the suitable media mix combination
* Analyzing and interpreting post-campaign results to evaluate the advertising performance and recommend future refinements

**Lydia Papanikou**

Graduate of Computer Engineering & Informatics at University of Patras

Business/System Analyst in Intrasoft International

* Working closely with clients such as the European Customs Development project (CUSDEV)
* Gathering customer requirements concerning critical messages exchange between EU customs to map system functionalities to the business system operations.
* Evaluate, analyze, and communicate systems requirements, including the delivery of monthly status reports

# Bibliography

MCLA Class Notes

Textbooks:

Deep Learning with Python, François Chollet

Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow-2nd Edition, Aurélien Géron

Links:

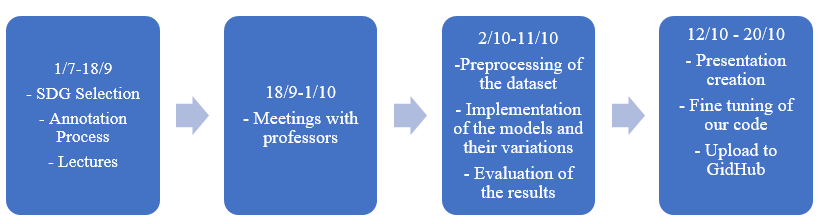
[https://keras.io](https://keras.io/)

[https://machinelearningmastery.com](https://machinelearningmastery.com/)

<https://www.kdnuggets.com/2020/03/tensorflow-keras-tokenization-text-data-prep.html>

<https://www.un.org/esa/desa/papers/2019/wp159_2019.pdf>

# Time Plan



# Contact Persons

Efstratios Marinos

Mobile: 6971883409

[stratos.marin@gmail.com](mailto:stratos.marin@gmail.com)

Lianos Alexandros

Mobile: 6945262871

[alexlianos88@hotmail.com](mailto:alexlianos88@hotmail.com)

Papanikou Lydia

Mobile: 6973680757

[lidiapapanikou94@gmail.com](mailto:lidiapapanikou94@gmail.com)

# Comments/Problems

The difficult part was to decide which type of Neural Network we should deploy in order to solve the problem with the most effective way. Because there is no clear rule of thumb everything must be tested to see which works best.

Regarding teamwork experience, different team members contribute different perspectives, and the synergy between team members can produce creative and productive results. Through conversations and controversies about each aspect of the project, we end up with a solution that make our project functional and output good quality results.

Another problem that we faced was that our data has imbalanced classes. Outside the purposes of this project, we should probably have collected more data so that a larger dataset was to be created. A larger dataset might expose a different and perhaps more balanced perspective on the classes.