Informatics Institute of Technology

In Collaboration With

The University of Westminster, UK



*The University of Westminster, Coat of Arms*

**6COSC020C Applied AI**

**Coursework**

Mr. Ammar Raneez

W1761196 | 2019163

Contents

[1. Part A 1](#_Toc117658686)

[Literature Review 1](#_Toc117658687)

[1. Part B 1](#_Toc117658688)

[ML algorithms 1](#_Toc117658689)

[SVM 2](#_Toc117658690)

[Multinomial Naïve Bayes 2](#_Toc117658691)

[Deep sequence models 2](#_Toc117658692)

[CNN (1D Convolution) 3](#_Toc117658693)

[RNN (LSTM) 3](#_Toc117658694)

[References 5](#_Toc117658695)

**Acronyms**

|  |  |
| --- | --- |
| RCT | Randomized Control Trial |
| SVM | Support Vector Machine |
| CNN | Convolutional Neural Networks |
| RNN | Recurrent Neural Networks |
| LSTM | Long Short-Term Memory |
| TF-IDF | Term Frequency-Inverse Document Frequency |

**Domain**: **hospitals & medicine**

**Problem Overview**: The [PubMed RCT dataset](https://arxiv.org/abs/1710.06071) consists of over 200,000 medical paper abstracts, some of the abstracts within the dataset are not structured well, therefore is hard to read.

**Goal:** The goal of this project is to create a model capable of splitting apart sentences in the abstract into structured sections (Ex: objectives, methods, results) to facilitate faster reading for researchers, in this case, in the field of medicine as abstracts can be especially long.

# **1. Part A**

## Literature Review

# **1. Part B**

Techniques that can be applied to the problem

* ML algorithms
  + Multinomial Naïve Bayes
  + SVM
* Deep sequence models
  + CNN (Convolutional Neural Network)
  + RNN (Recurrent Neural Network)

## ML algorithms

ML algorithms do not understand text; therefore, the data must be converted into numbers. Specifically, in this case, the labels would be the category each sentence belongs to, and the features would be the sentence itself. The labels can be encoded into numbers and the sentences be passed into a [TFidfVectorizer](https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html) that will convert the sentences to numbers. The encoded labels and vectorized sentences can then be passed into the naïve bayes and SVM algorithms as inputs. The output is expected to return a list of encoded labels upon evaluating with some test sentences.

### Multinomial Naïve Bayes

The naïve bayes is simply a classifier based on applying the statistical Bayes’ theorem. A multinomial naïve bayes model can be used to fit the data using a TF-IDF algorithm. In the [Scikit-Learn ML roadmap](https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html), where the Naïve Bayes algorithm is what is recommended as a baseline starting point for textual data.

The main advantages of this algorithm are that it is simple to implement, really scalable and can handle large amounts of data, and is therefore, quick to train. However, the drawback is that this algorithm can produce relatively lower accuracy in comparison to more powerful ones. Furthermore, although the algorithm is considered to be a good classifier, it is not a great estimator as it assumes that all the features are independent.

Since this algorithm does not perform well, it will not be the author’s choice for implementation, however, it can be used as a baseline to compare the main model against.

### SVM

SVMs are a set of supervised learning methods used for classification, regression and outlier detection. The SVM model can also be used to fit the data using the TF-IDF algorithm as mentioned.

The main advantages of SVMs are that they are effective when the data is in high dimensions, memory efficient since they can utilize a subset of data at a time, and versatile as there are multiple functions that can be specified for the SVM to take a decision. However, they are not suitable for larger datasets and data that contains so much noise.

Since the PubMed dataset is quite large and very likely to contain noise, the author will not utilize the SVM for implementation.

## Deep sequence models

Inputs for these models are similar to the ones required by the multinomial naïve bayes. A vectorization and an embedding layer will be required to convert the text to numbers and to capture relationships between those numbers respectively.

Furthermore, the models work best when all the sentences are of the same length as creating batches will be facilitated; which can be implemented by further adding a tokenization layer.

Therefore, the sentences would be vectorized and embedded beforehand. Furthermore, the labels must be one hot encoded, as TensorFlow’s loss function prefer one hot encoded data. Ultimately the sentences would be vectorized and embedded, while the labels would be one hot encoded, and those would be fed into the model for training.

Upon training, the outputs of the models would be an array of prediction probabilities upon evaluating on the test data where the predicted category of the sentence would be the index at with the highest probability.

### CNN (1D Convolution)

A 1D convolutional can be used to model sequence data. The CNN was initially developed for image classification problems, where the model can learn the internal representations of a 2D input; however, this same procedure can be tweaked to work on 1D sequence data, hence the term Conv1D.

The drawbacks of CNN are that since they are complicated networks, they require a large amount of data and as a result takes much longer time to train. Furthermore, they can get affected by class imbalance and could tend to be biased to the most occurring class. The main advantage of the CNN is that they can perform quite well and at times even match the performance of RNNs, they have better parallelism and can exhibit more stable gradients.

However, as mentioned, the CNN is mainly tailored for image data, and there is a better option for the modelling of sequence data: RNN.

### RNN (LSTM)

An RNN processes sequence input by iterating through the elements, and pass the outputs from one timestep to the next. A further Bidirectional RNN can also pass the data backwards to the previous timestep. Arguably, the most common RNN for sequence modelling is the LSTM.

LSTMs are great for sequence modelling, especially since it has memory, which is essential when modelling data where the context is important, specifically in this case, for sentences, as the following word depends on the previous. Additionally, training can be much faster as there is no need for parameter fine-tuning as they work well over a broad range of parameters. They do however have certain drawbacks. Since it is also a deep neural net, it takes longer time to train and can require more memory. Further, they can easily overfit the data.

**Choice of algorithm to explore further**

The author believes that the best way to implement the model is by utilizing an LSTM, since it has the capability of storing the memory of the past, where in this case specifically, have information of the words prior to the current in the specific sentence. Although the model could overfit, this can be solved using Regularization techniques if necessary, and Google Colab can be utilized to train the model.

# **References**

Naive Bayes Classifier: Pros & Cons, Applications & Types Explained. (2022). upGrad blog. Available from <https://prod-eks-app-alb-1037681640.ap-south-1.elb.amazonaws.com/blog/naive-bayes-classifier/> [Accessed 25 October 2022].

scikit-learn: machine learning in Python — scikit-learn 1.1.2 documentation. (no date). Available from <https://scikit-learn.org/stable/index.html> [Accessed 25 October 2022].

Tompson, J. et al. (2015). Efficient object localization using Convolutional Networks. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). June 2015. Boston, MA, USA: IEEE, 648–656. Available from <https://doi.org/10.1109/CVPR.2015.7298664> [Accessed 25 October 2022].

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9 (8), 1735–1780. Available from <https://doi.org/10.1162/neco.1997.9.8.1735> [Accessed 25 September 2022].

K, D. (2020). Top 4 advantages and disadvantages of Support Vector Machine or SVM. Medium. Available from <https://dhirajkumarblog.medium.com/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107> [Accessed 25 October 2022].