21AIE314 NATURAL LANGUAGE PROCESSING

TEAM-13PROJECT REPORT – CLINICAL TEXT CLASSIFICATION

MAY 4, 2022

ARUN PRAKASH	- CB.EN.U4AIE19014
C.R. ASSWIN	- CB.EN.U4AIE19016
DHARSHAN KUMAR	- CB.EN.U4AIE19024
AVINASH DORA	- CB.EN.U4AIE19028

TABLE OF CONTENTS:

I.ABSTRACT :

Word embedding motivated by deep learning has shown promising results over Bag-of -words features for natural language processing. When trained on large text corpora, word embedding methods such as TF-IDF, Glove, and word2vec methods have the advantage of learning from unlabeled data and reducing the dimension of the feature space. In this study, we experimented with TF-IDF and VMD features for a set of clinical text classification tasks and compared the results with Sequential Models such as LSTM and attension-based transformers. In this study, we have discussed a detailed comparison between the performance of sequence and non-sequence modeling.

II. INTRODUCTION :

Text classification, the process of assigning one or numerous categories from a set of options to a document, is a prominent and well-researched task in Natural Language Processing (NLP) and text mining. There exist different kinds of Text classification methods which include simple binary classification, multi-class classification, and multi-label classification. In clinical natural language processing, document classification is critical . When unstructured clinical notes are linked to structured data from electronic health records (EHRs), labels are typically only available at the document level rather than at the individual word level, making document categorization an important tool in the practical automation of clinical operations.

III.LITERATURE SURVEY

The need for text classification has become concrete across various domains such as categorization of mails and spams, classifying fake news from the real ones and to identity fake medical report from real ones. The use of convolutional neural networks to extract information from a given signal has opened transcended from image processing to feature extraction from text as well. The use of text-CNN requires the use of one-dimensional convolutional network. In [1] the authors study the effect of CNN for the task of text classification on the locale language dataset of Ethiopia. They also explored the use of different pre-trained word embedding architectures like skip-gram, word2vec, fast-text, CBOW with different CNN architectures. The proposed architecture using CBOW with word2vec outperformed all traditional machine learning models. Document classification is an essential task in clinical natural language processing. In the clinical setting, labels are assigned only based on document level rather than individual word level. Efficient and robust classification of elements from the clinical documents is extremely important for applications such as precision medicine, population health surveillance, and research. Unfortunately, in the clinical setting, human annotation of electronic health records can be extremely time-consuming and expensive due to the technical nature of the content and the expert knowledge required to parse it.

Medical text classification has an undoubted effect on the way diseases are being diagnosed and how the solutions are being offered in computer aided diagnoses models. Since the data considered is never similar between any two medical practitioners and as medical prescriptions has its own intricacies, medical text classification is challenging. In addition to the above challenges, it contains terminologies that describe medical concepts and terminologies. The above-mentioned challenges initiate the need for a word level document classifier using health records.

Bidirectional Encoder Representations from Transformers (BERT) based approaches are the current state-of-the-art in many natural language processing tasks; however, their application to document classification on long clinical texts is limited. Gao et al. [2] introduced the limitations of the BERT based approach and simpler word level CNN and a hierarchical self-attention network. All experiments were performed on the MIMIC-III dataset for comparison. BERT being a computationally expensive deep learning approach, utilizes subword-level WordPiece tokens rather that word-level tokens as input exhibits non-trivial challenges. While BERT is pre-trained on 512 WordPiece which is equivalent to 400 words, clinical transcriptions can easily exceed this limit. The unavailability of open-sourced pre-trained models in this clinical data domain limits its ability as a transfer learning tool. The HiSAN network is based on self-attention operations with the advantage of having lesser number of parameters that the BERT model. The HiSAN's lower hierarchy uses a series of attention-based operations to generate a fixed-length vector representations. Attention based operations is used following the upper hierarchy to generate vector representations of the documents. This document vector is passed to a final dense classification layer that uses softmax for single-label classification and sigmoid for multilabel classification. Yao et al. [3] proposes an approach based on combines rule-based features and knowledge-guided deep learning models for effective disease classification. Critical steps of their method include recognizing trigger phrases, predicting classes with very few examples using trigger phrases and training a convolutional neural network (CNN) with word embeddings and Unified Medical Language System (UMLS) entity embeddings. The proposed architecture was evaluated on the i2b2 obesity challenge. Yahia et al. [4] reviewed all the existing CNN based medical text classification approaches and results. Yoon et al. [5] introduce graph-of-words and graph-based convolution neural network technique, the Text Graph Convolutional Network (Text GCN), in the context of performing classification tasks on free-form natural language texts. They designed a study of multi-task information extraction from medical text documents and implemented multi-task learning in the Text GCN, performed hyperparameter optimization, and measured the clinical task performances. The study was evaluated on micro and macro-F1 scores of four information extraction tasks, including subsite, laterality, behaviour, and histological grades from cancer pathology reports.

IV. METHODOLOGY :

4.1 DATA PREPARATION

From the counts of medical speciality column, imabalance of data can be obererved, to alleviate this issue , data samples with value count less than 100 is removed. In addition to this SMOTE algorithm is used inorder to improve the data balance after text encoding.

4.2 DATA PREPROCESSING

4.2.1 Handling Missing data:

Generally missing data is not allowed in machine learning.Missing data can be dropped if the number of missing entries is relatively small. The other common method in handling missing data in NLP is to copy information from other related feature in the dataset. For example the missing entries in the NLP feature **'transcription**' can be filled up by copying the corresponding entries from the feature 'description' because the description is a summary of the transcription. In this experiment, missing entries in the feature transcription column is dropped.After dropping there are a total of 33 missing entries.

4.2.2 Extract Medical Entities

The most important contents in the feature column transcription are medical notes.It is used for extracting medical entities for clinical text classification. This is achieved by using spacy library. This library contains spaCy models for processing biomedical, scientific or clinical text.In this project **'en_ner_bionlp13cg_md'** model is used. It is trained on the **BIONLP13CG** corpus.

ENTITY TYPES OF MODEL	AMINO_ACID, ANATOMICAL_SYSTEM, CANCER, CELL, CELLULAR_COMPONENT, DEVELOPING_ANATOMICAL_STRUCTURE, GENE_OR_GENE_PRODUCT, IMMATERIAL_ANATOMICAL_ENTITY, MULTI-TISSUE_STRUCTURE, ORGAN, ORGANISM, ORGANISM_SUBDIVISION, ORGANISM_SUBSTANCE, PATHOLOGICAL_FORMATION, SIMPLE_CHEMICAL, TISSUE

4.2.3 Standardize Text

Two common steps to standardize text are changing text to lower case which helps to avoid the effort for ML models to learn the case of characters and next step is removing punctuation which helps in reducing the size of the vocabulary.

4.2.4 Tokenize Text :

This helps to divide a text into elements of vocabulary. Generally,word-level (1-gram) tokenization is used if text is treated as sequence of words. Word-level tokenization also enables the activities of removing stopwords, stemming, and lemmatization.After Tokenizing the text, stopwords are removed and stemming and lemmatization is applied inorder to reduce the size of the vocubulary.

4.3 FEATURE ENGINEERING

Once a text feature is preprocessed, each text (document) needs to be encoded as a vector of numbers (known as vectorization) since MLand DL models only understand numbers. The selection of encoding methods depends on how text is interpreted. In text classification, a text string (document) is typically treated as either a bag of words or a sequence of words. The following are some of the common text encoding/vectorization methods.

4.3.1 Word Embedding Techniques

Word embedding encodes the meaning of the word in such a way so that the words that are closer in the vector space are similar in meaning. In this technique each word tries to capture a linguistic meaning(describing whether that particular word is verb or entity or noun or person etc). initially these weights(features) are randomized and during training process it gets updated and same meaning words/vectors comes close together.

TF-IDF

TF-IDF vectorization encodes one text/document as a sparse vector of the length of vocabulary. In TF-IDF encoded vector contains not only the presence information of the corresponding words/tokens in the vocabulary but also the token frequency (TF) weighted by the inverted document frequency (IDF). Because of this, a TF-IDF encoded vector has more prediction power than some of other techniques such as a Muti-Hot encoded vector.Along with TF-IDF ,PCA(Principal Component Analysis) is applied in this project indorder to reduce the dimensionality.

POSITIONAL EMBEDDING

Word order information plays an important role in text classification. Positinal embedding is done to add word information into word embedding to increase the performance of text classification using transformers. In this technique both meaning and postion of word is captured.

Since LSTM takes the word embedding sequentially , they know the order in which the words appeared. But transformers on the other hand takes all the input embeddings all together.Hence they tend to loose the ordering.As position place an important role in NLP tasks, it should be given more impotance.Therefore inoder to capture the posiitonal information, new postional embedding vectors is introduced known as positional embeddings. Word embedding vector and positional embedding vector are combined to produce a new vectors which contains both the information.

Sine-cosine similarity is most commonly used to capture the position embedding in an effiencent manner At odd positions sine formula is used to get its corresponding position embedding, similary at even positions, cosine formula is applied.

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d}}}\right) \qquad PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{\frac{2i}{d}}}\right)$$

Here “pos” refers to the position of the “word” in the sequence. “i” refers to index of the position embedding vector and “d” represents the dimension.

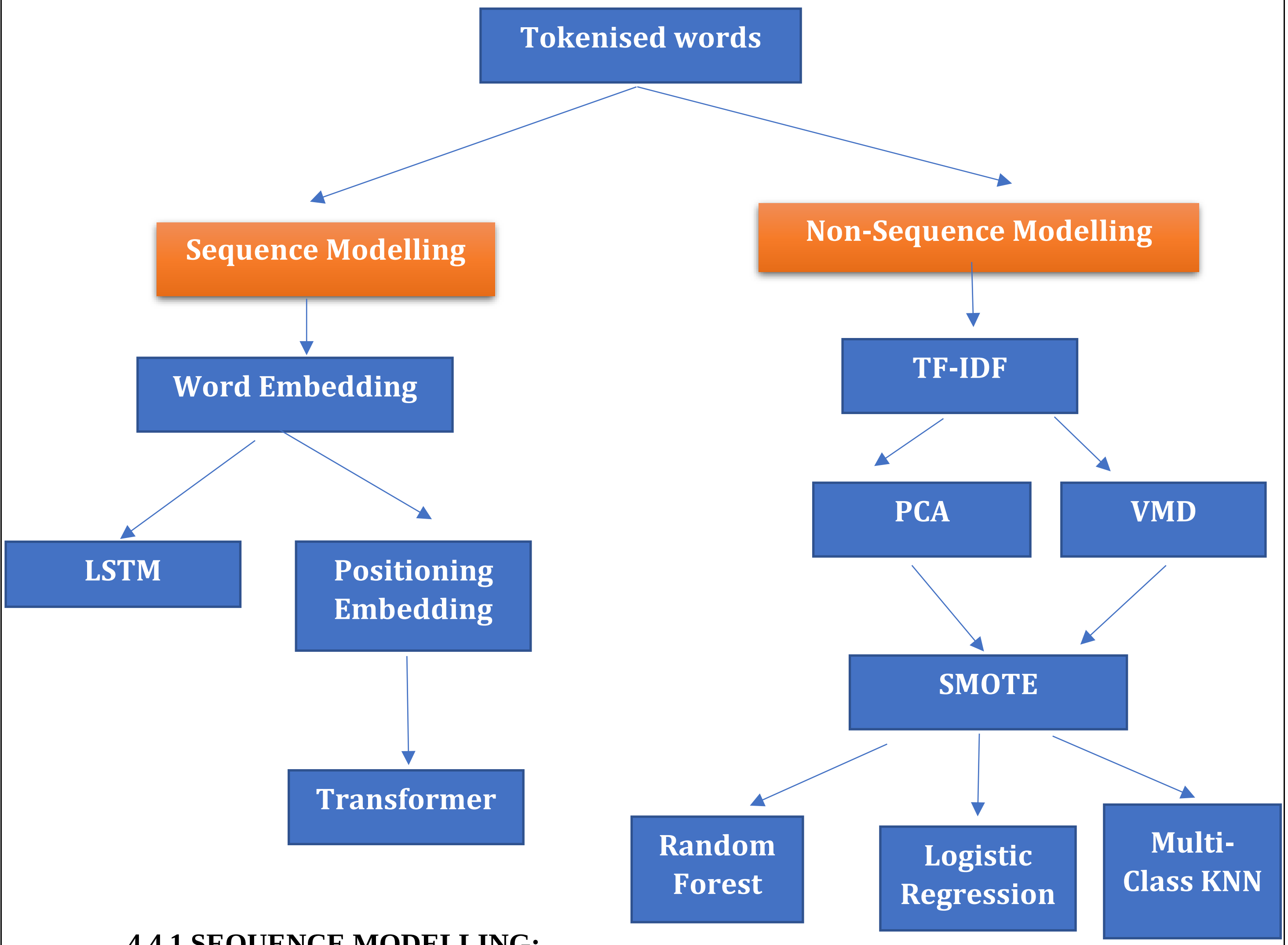
4.4 MODEL SELECTION

Once the dataset is prepared for modeling, we need to select a machine learning/deep learning model and train it with the prepared data.

Similarly to the selection of text encoding methods, the selection of a machine learning or deep learning model for text classification depends on how text is encoded. Specifically, we can select a model as shown in the table below:

Text Encoding Method	**ML/DL Models**
TF-IDF	Any Non- Sequence classification Model – Logistic Regression, Random Forest, multi-class classifier
Word Embedding	Any Sequence classification model - **LSTM** , GRU
Positional Embedding	**Transformers**

Pipeline :



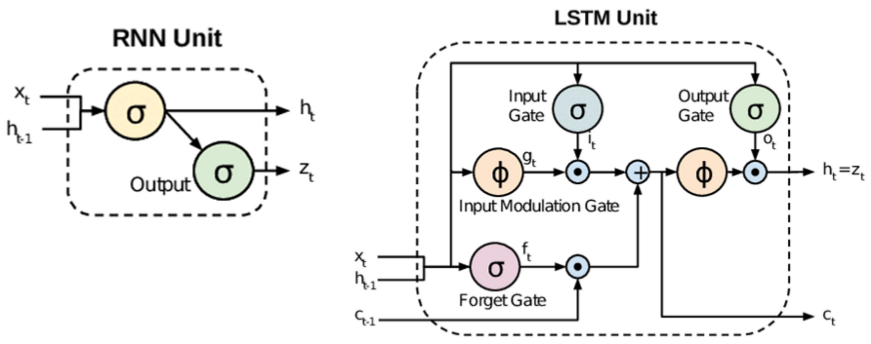
4.4.1 SEQUENCE MODELLING:

Sequence models are the machine learning models that input or output sequences of data. Sequential data includes text streams, audio clips, video clips, time-series data and etc. RNN, LSTM, transformers are commonly used seqence modeling in NLP domain. In this we have analysed sequence modeling for LSTM and Transformers.

LSTM :

Long Short-Term Memory was introduced by Sepp Hochreiter and Juergen Schmidhuber. LSTM is a modification to the RNN hidden layer. LSTM has enabled RNNs to remember its inputs over a long period of time. In LSTM in addition to the hidden state, a cell state is passed to the next time step.

LSTM can capture long-range dependencies. It can have memory about previous inputs for extended time durations.

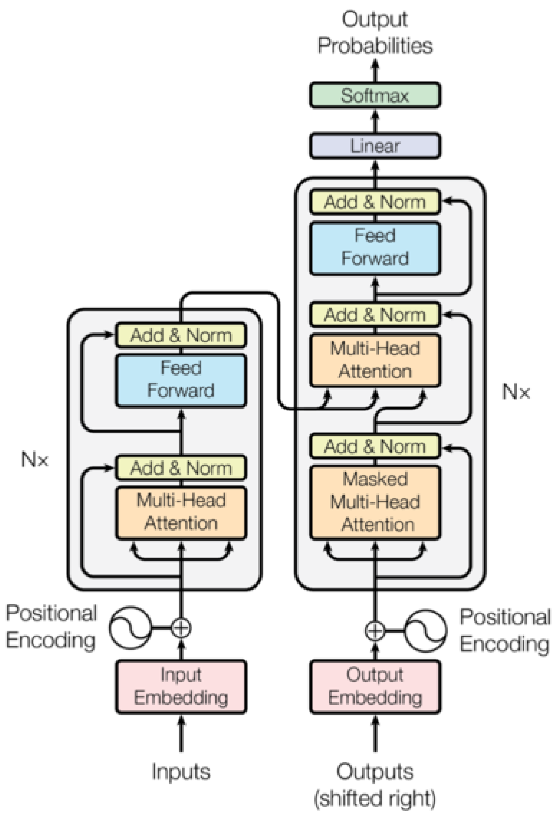


There are 3 gates in an LSTM cell. Memory manipulations in LSTM are done using these gates. Long short-term memory (LSTM) utilizes gates to control the gradient propagation in the recurrent network's memory.

- Forget Gate: Forget gate removes the information that is no longer useful in the cell state.
- Input Gate: Additional useful information to the cell state is added by input gate.
- Output Gate: Additional useful information to the cell state is added by output gate.

Transformers :

Recently the attention-based Transformer algorithm becomes a preferred algorithm over LSTM for sequence modeling.



ENCODER

Generally the encoder consists of a stack of 6 identical layers, where each layer is composed of two sublayers:

1. The first sublayer implements **a multi-head self-attention** mechanism. We had seen that the multi-head mechanism implements h heads that receive a (different) linearly projected version of the queries, keys and values each, to produce h outputs in parallel that are then used to generate a final result.
2. The second sublayer is a fully connected feed-forward network, consisting of two linear transformations with Rectified Linear Unit (ReLU) activation in between:

$$\boxed{\mathrm{FFN}(x) = \mathrm{ReLU}(W_1 x + b_1)W_2 + b_2}$$

The six layers of the Transformer encoder apply the same linear transformations to all of the words in the input sequence, but each layer employs different weights and bias parameters to do so. Furthermore, each of these two sublayers has a residual connection around it. Each sublayer is also succeeded by a normalization layer, layernorm, which normalizes the sum computed between the sublayer input, x, and the output generated by the sublayer itself, sublayer(x).

DECODER

The decoder shares several similarities with the encoder. It also consists of a stack of N = 6 identical layers that are, each, composed of three sublayers:

The first sublayer gets the decoder stack's previous output, adds positional information, and applies multi-head self-attention to it. While the encoder is meant to pay attention to all words in the input sequence, regardless of their location, the decoder is adjusted to only pay attention to the words that come before them. As a result, the prediction for a word at position I can only be based on the known outputs for the words in the sequence before it. This is accomplished in the multi-head attention mechanism (which implements several, single attention functions in simultaneously) by applying a mask on the values obtained by scaling matrices Q and K. Masking is done by suppressing matrix values that would otherwise correspond to illegal connections. The second layer of the encoder utilises a multi-head self-attention technique identical to the one used in the first sublayer. This multi-head mechanism gets the queries from the preceding decoder sublayer, as well as the keys and values from the encoder output. The decoder can now focus on all of the words in the input sequence. A fully connected feed-forward network, similar to the one established in the encoder's second sublayer, is implemented in the third layer. On the decoder side, the three sublayers have residual connections around them and are followed by a normalisation layer.

4.4.2 NON-SEQUENCE MODELLING

Unlike sequential data, in non-sequential modelling, the order of sequence is not taken into consideration and for noraml machine learning algorithms such as Logistic regression, random forest, Multi class output classifier is used this project for prediction.

PCA :

Principal Component Analysis is an unsupervised learning approach used in machine learning to reduce dimensionality. It is a statistical procedure that uses orthogonal transformation to turn correlated feature observations into a set of linearly uncorrelated features. PCA generally tries to find the lower-dimensional surface to project the high-dimensional data. PCA decreases dimensionality by evaluating the variance of each characteristic because the high attribute indicates a fair split between the classes.

Logistic Regression

The Supervised Learning technique includes one of the most common Machine Learning algorithms: logistic regression. It's used to predict a categorical dependent variable from a group of independent variables. A categorical dependent variable's output is predicted using logistic regression. As a result, the result must be a discrete or categorical value. It can be Yes or No, 0 or 1, true or false, and so on, but instead of giving exact values like 0 and 1, it delivers probabilistic values that are somewhere between 0 and 1.**Random Forest:**

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

Multi-Output Classifier with k-NN

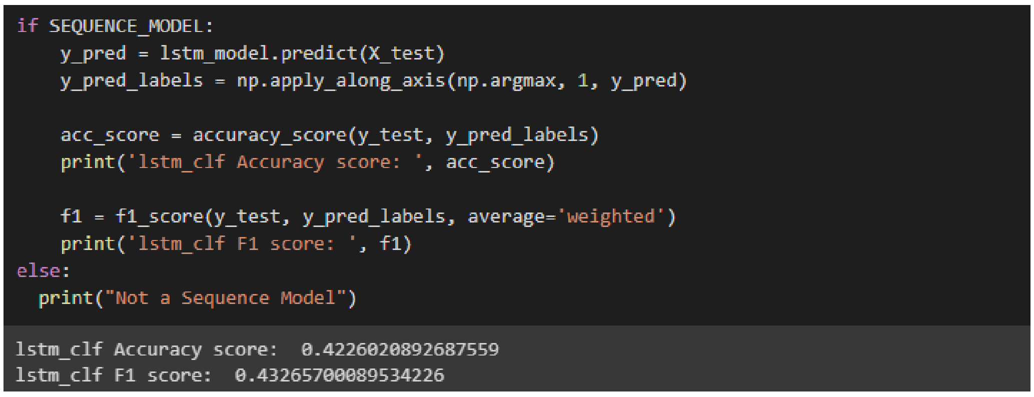
This strategy consists of fitting one classifier per target. This is a simple strategy for extending classifiers that do not natively support multi-target classification.

4.5 MODEL TRAINING AND EVALUATION

Genrally dataset is split into training, validation and test or simplicity, the dataset is only split into two subsets in this project, one for model training and the other for model testing.in our experiment the split ratio of train and test is taken as 75:20 respectively.

4.5.1 Model Evaluation(Sequence Modelling)

- **LSTM :**



```
if SEQUENCE_MODEL:
    y_pred = lstm_model.predict(X_test)
    y_pred_labels = np.apply_along_axis(np.argmax, 1, y_pred)

    acc_score = accuracy_score(y_test, y_pred_labels)
    print('lstm_clf Accuracy score: ', acc_score)

    f1 = f1_score(y_test, y_pred_labels, average='weighted')
    print('lstm_clf F1 score: ', f1)
else:
  print("Not a Sequence Model")
```

```
lstm_clf Accuracy score:  0.4226020892687559
lstm_clf F1 score:  0.43265700089534226
```

- **Confusion Matrix :**

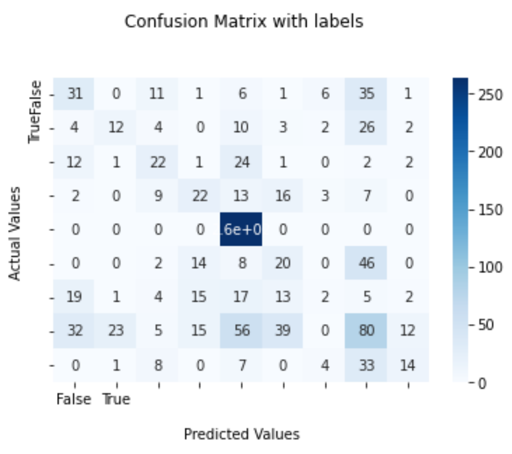
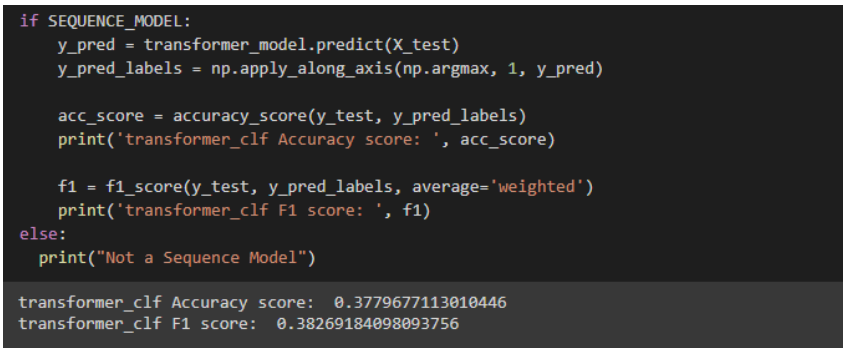


Fig Confusion Matrix for LSTM model

- **TRANSFORMERS :**



```
if SEQUENCE_MODEL:
    y_pred = transformer_model.predict(X_test)
    y_pred_labels = np.apply_along_axis(np.argmax, 1, y_pred)

    acc_score = accuracy_score(y_test, y_pred_labels)
    print('transformer_clf Accuracy score: ', acc_score)

    f1 = f1_score(y_test, y_pred_labels, average='weighted')
    print('transformer_clf F1 score: ', f1)
else:
  print("Not a Sequence Model")
```

```
transformer_clf Accuracy score:  0.3779677113010446
transformer_clf F1 score:  0.38269184098093756
```

- **Confusion Matrix:**

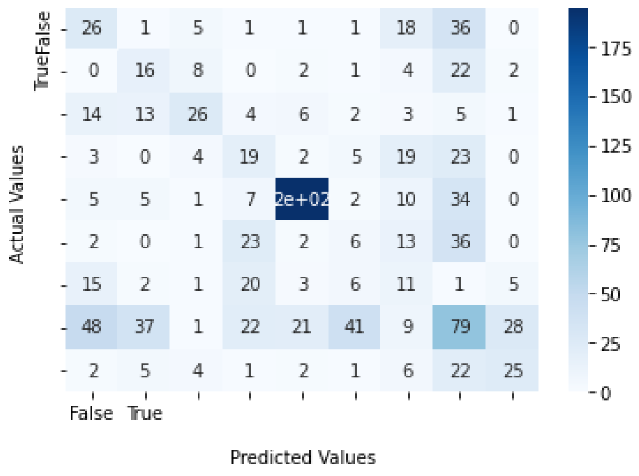
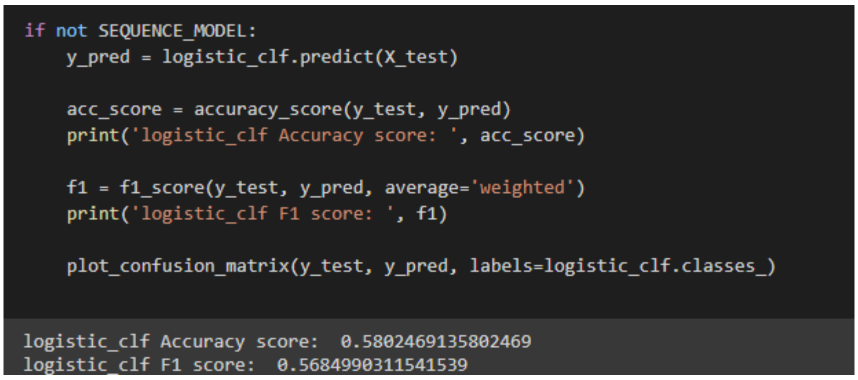


Fig Confusion Matrix for attension-basedTransformers

4.5.2 Model Evaluation(Non- Sequence Modelling)

- **Logistic Regression**



```
if not SEQUENCE_MODEL:
    y_pred = logistic_clf.predict(X_test)

    acc_score = accuracy_score(y_test, y_pred)
    print('logistic_clf Accuracy score: ', acc_score)

    f1 = f1_score(y_test, y_pred, average='weighted')
    print('logistic_clf F1 score: ', f1)

    plot_confusion_matrix(y_test, y_pred, labels=logistic_clf.classes_)
```

```
logistic_clf Accuracy score:  0.5802469135802469
logistic_clf F1 score:  0.5684990311541539
```

- **Confusion Matrix :**

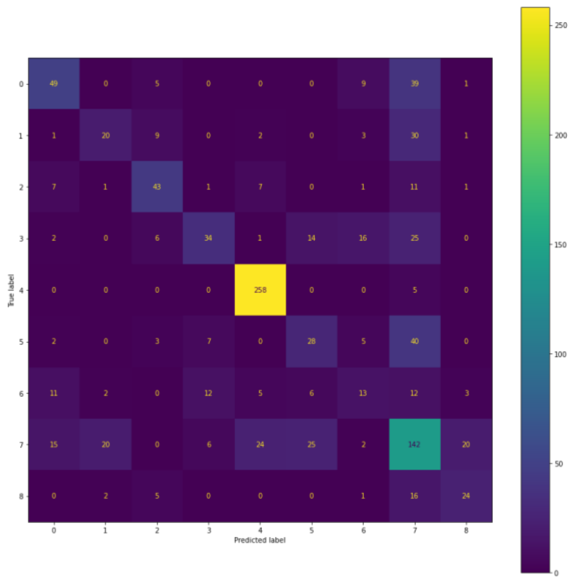
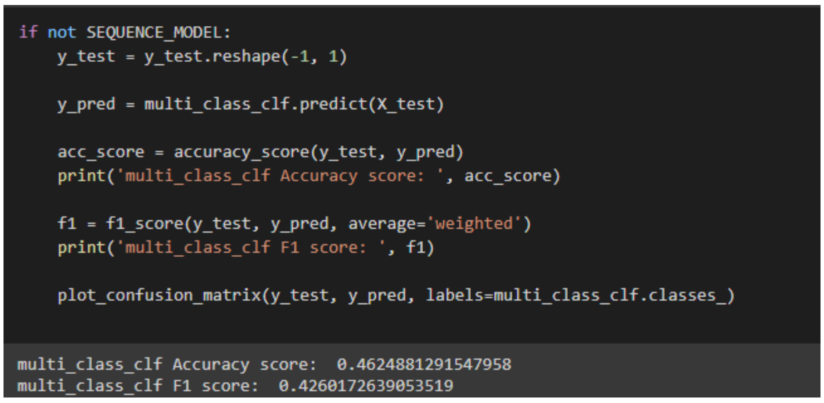


Fig Confusion Matrix for Logistic regression

- **Multiclass Classifier**



```
if not SEQUENCE_MODEL:
    y_test = y_test.reshape(-1, 1)

    y_pred = multi_class_clf.predict(X_test)

    acc_score = accuracy_score(y_test, y_pred)
    print('multi_class_clf Accuracy score: ', acc_score)

    f1 = f1_score(y_test, y_pred, average='weighted')
    print('multi_class_clf F1 score: ', f1)

    plot_confusion_matrix(y_test, y_pred, labels=multi_class_clf.classes_)
```

```
multi_class_clf Accuracy score:  0.4624881291547958
multi_class_clf F1 score:  0.4260172639053519
```

- **Confusion Matrix :**

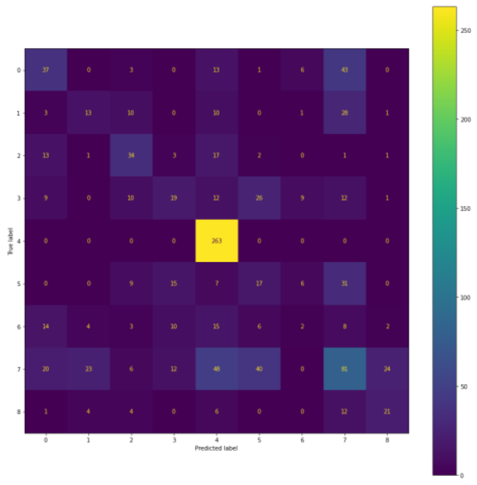
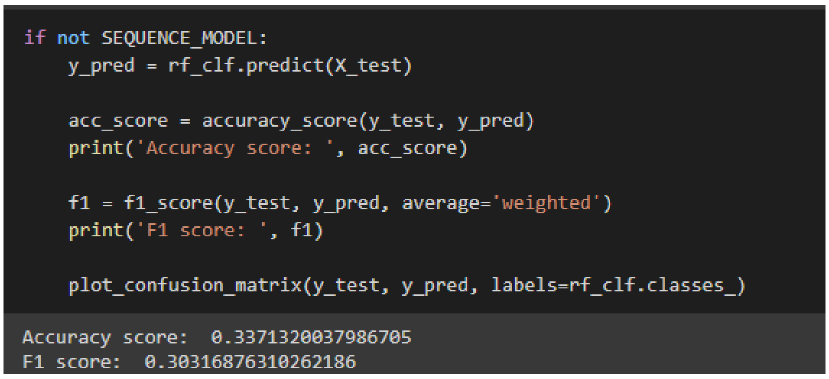


Fig Confusion Matrix for Multi-class classifier-knn

- **Random Forest Classifier :**



```
if not SEQUENCE_MODEL:
    y_pred = rf_clf.predict(X_test)

    acc_score = accuracy_score(y_test, y_pred)
    print('Accuracy score: ', acc_score)

    f1 = f1_score(y_test, y_pred, average='weighted')
    print('F1 score: ', f1)

    plot_confusion_matrix(y_test, y_pred, labels=rf_clf.classes_)
```

```
Accuracy score:  0.3371320037986705
F1 score:  0.30316876310262186
```

- **Confusion Matrix :**

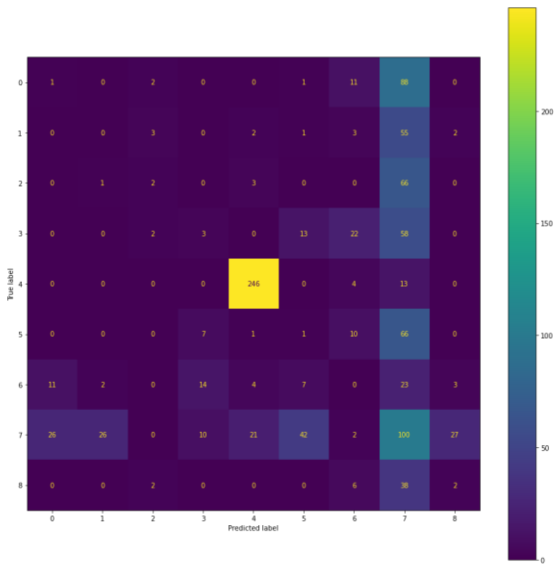


Fig Confusion Matrix for random forest classifer

V. DATASET:

All the experiments in this study were conducted on an open-source clinical text dataset from Kaggle titled " mtsamples.csv". The dataset contains 5000 rows and 5 columns. Description, medical_speciality, sample_name, and transcription keywords are the columns present in the dataset. For the experiment, transcription is considered as a feature column and medical specialty is considered as the target label. There are a total of 40 unique categories present in the medical specialty column. After filtering and removing unrelated categories we arrive at 9 categories.

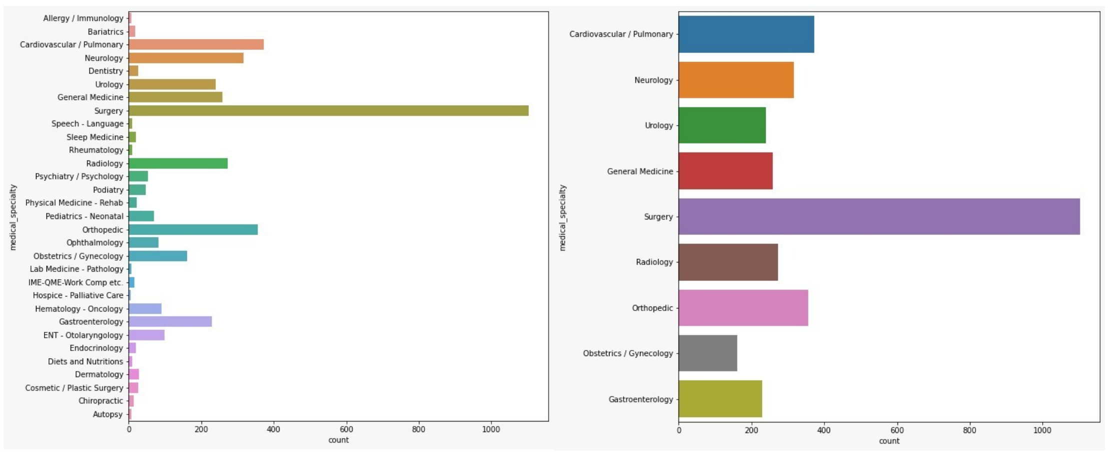


Fig.Plots representing categories after and before preprocessing

VI. RESULTS AND DISCUSSION:

TABLE

Model	**Accuracy**	**F1-Score**
Logistic Regression	0.5802	0.5684
Multi-Output Classifier	0.4624	0.4260
Random Forest	0.3371	0.3031
LSTM	0.4226	0.4326
Transformers	0.3912	0.4028

From the above results table we can observe that Logistic regression,Multi class output classifier and LSTM are top 3 performing model with 58.02%,46.24%,42.26% accuracy respecrively.There reason for the non-sequence models to perform relatively better when comapared to sequence modelling is because of the type of dataset, as majority of the words in the trancription coloumn are medical based and those words are not depends upon the previous sequence.From the confusion matrix of all the trained models we can observe that logistic regression, multiclass classifier, and transformers are able to predict the class labels on the test data.On the other hand there is missclassification in LSTM and Random forest classification.

VII. CONCLUSION & FUTURE WORKS :

In this project we have implemented classification for clinical text using sequence and non-sequence modeling, In sequence modeling, LSTM and Transformers models use word embedding and positional word embedding respectively and in non-sequence modeling, ML models such as Logistic regression, multi-class classifier with knn and the random forest is used. From the results and discussion section, some models are not able to classify the categories properly.

To improve these results as future work, in sequential models Bio BERT(Bidirectional Encoder Representations from Transformer) can be applied as it understands the meaning of ambiguous language in the text by using surrounding text to establish context. Also in nonsequence modeling, instead of using ML models individually, using stacking classifiers(stacking different ML models as base and meta classifiers) to improve accuracy and other metrics.

VIII.REFERENCES :

[1] Fesseha, Awet, et al. "Text classification based on convolutional neural networks and word embedding for low-resource languages: Tigrinya." Information 12.2 (2021): 52.

[2] Gao, Shang, et al. "Limitations of transformers on clinical text classification." IEEE journal of biomedical and health informatics 25.9 (2021): 3596-3607.

[3] Yao, Liang, Chengsheng Mao, and Yuan Luo. "Clinical text classification with rule-based features and knowledge-guided convolutional neural networks." BMC medical informatics and decision making 19.3 (2019): 31-39.

[4] Yahia, Hazha Saeed, and Adnan Mohsin Abdulazeez. "Medical text classification based on convolutional neural network: A review." International Journal of Science and Business 5.3 (2021): 27-41.

[5] Yoon, Hong-Jun, et al. "Information extraction from cancer pathology reports with graph convolution networks for natural language texts." 2019 IEEE International Conference on Big Data (Big Data). IEEE, 2019.

[6]. https://s3-us-west-2.amazonaws.com/ai2-scispacy/releases/v0.5.0/en_ner_bionlp13cg_md-0.5.0.tar.gz(medical NER model en_ner_bionlp13cg_md)

[7]. https://towardsdatascience.com/introduction-to-sequence-modeling-problems-665817b7e583

[8]. https://datascience.stackexchange.com/questions/51065/what-is-the-positional-encoding-in-the-transformer-model.

[9]. https://machinelearningmastery.com/the-transformer-model/

[10]. F. Sebastiani, “Machine learning in automated text categorization,” ACM computing surveys (CSUR), vol. 34, no. 1, 2002, pp. 1–47.