An Overview of Text Summarization Techniques

Divyansh Mulchandani
SCOPE, Vellore Institute of Technology, Vellore

Debangan Mandal
SCOPE, Vellore Institute of Technology, Vellore

Aileni Rohan Reddy
SCOPE, Vellore Institute of Technology, Vellore

Abstract

Text Summarization is the process of creating a condensed form of text document which maintains significant information and the general meaning of source text. Automatic text summarization becomes an important way of finding relevant information precisely in large text in a short time with little effort. Text summarization approaches are classified into two categories: extractive and abstractive. This paper presents a comprehensive survey of both approaches in text summarization.

Keywords: Text Summarization, Natural Language Processing, Extractive Summary, and Abstractive Summary.

Problem Description

As the amount of information on the internet is increasing rapidly day by day in different formats such as text, codes, images, audio, and video. It has become difficult for an individual to find relevant information of his interest. Suppose user queries for information on the internet he may get thousands of result documents that may not necessarily be relevant to his concern. So, there is a problem of searching for relevant documents from the number of documents available and absorbing relevant information from it. To solve the above two problems, automatic text summarization is very much necessary.

Objective

To find appropriate information, a user needs to search through the entire documents this causes information overload problems which lead to wastage of time and effort. So, there is a problem of searching for relevant documents from the number of documents available and absorbing relevant information from it. For this, automatic text summarization aims to become an important way of finding relevant information precisely in large text in a short time with little effort. Thus, the objective or the purpose of this paper presents the comprehensive survey of both the approaches (extractive summarization and abstractive summarization) in text summarization.

Introduction

1. Significance of the problem

Summarization systems often have additional evidence they can utilize to specify the most important topics of a document. For example, when summarizing blogs, discussions or comments are coming after the blog post that is a good source of information to determine which parts of the blog are critical and interesting.

In scientific paper summarization, there is a considerable amount of information such as cited papers and conference information which can be leveraged to identify important sentences in the original paper.

This also has a major application in academic, research, medical, journalism, and literature, especially to have a manual analysis of several documents in these domains.

2. Intended Audience and Readings

The intended audience for this design is the scientific community responsible for the research in AI applications and hardware, AI developers, AI performance and security analysts, System and Software Engineers, and everyone interested in Computer Science, IT, and Computer Engineering.

We would suggest the stakeholders read up on frameworks for AI using Python and the testing with benchmarks matrix structure of AI before going through the document further.

3. System Overview Background

Before we move on to text summarization, we must first understand what a summary is. A summary is a text created from one or more texts that communicate key information from the original material in a condensed form and remove unnecessary tokens.

The text summarization process works in three steps: analysis, transformation, and synthesis. Analysis step analyses source text and select attributes. The transformation step transforms the result of the analysis and finally, representation of summary is done in the synthesis step. As a result, the purpose of automatic text summarizing is to offer the original material in a condensed form with semantics.

4. Applications

Automatic summarization condenses a source document into meaningful content which reflects the main thought in the document without altering information. Thus, it helps the user to grab the main notion within a short period. If the user gets an effective summary it helps to understand the document at a glance without checking it entirely, so time and effort could be saved.

5. Motivation

Text summarization is the process of identifying the most relevant information in a document or set of related documents and conveying it in a shorter span than the original text. Some of the reasons for motivation of text summarization are as follows:

a. To get the summarized teaching information from the transcript for the student.
b. To get the summarized meeting information from the transcript for the business operations or meeting held for other purposes.
c. With summaries of the meeting's transcript, people, especially higher-level authorities, can make effective decisions in less time.
d. With summaries of transcripts of people's calls and messages, it is helpful for the crime department's executive to classify whether there were any illegal activities.
e. Automatic summarization improves the effectiveness of indexing in a transcript.

Relevance of the Problem

A summary is a text that is produced from one or more texts, that conveys important information in the original text and is of a shorter form. The goal of automatic text summarization is to present the source text in a shorter version with semantics. The most important advantage of using a summary is, it reduces the reading time. The typical length of summarization should be 5 to 10 percent of the main text, without losing its semantics/meanings. So, the study of evaluations made by the machine learning algorithm in automatic text summarizer influences the outcome through the followings aspects:

1. Ideal length of the summary.
2. Text Data in different sections or headings of summary.
3. Semantics [meaning] incorporated in the summary.

Expected Outcomes

This technical report of the survey paper covers both extractive and abstractive summarization techniques. The first method uses the standard statistical technique of tokens of words in a sentence to rank sentences and relevance, while the second method uses the latent semantic analysis technique using machine learning to identify semantically important sentences, for summary creations. This is an attempt to study the techniques for creating a summary with wider coverage of the document's content and less redundancy. Despite the very different approaches taken by the two summarizers, they may both produce quite quality summarization and high-performance scores.

Background

1. Extractive Summarization

In extractive summarization, a summary from the given text is created by selecting a subset of the total sentence base. Most important phrases or sentences from the text are identified and selected based on a score that is computed depending on the words in that sentence.

2. Abstractive Summarization

In the method of abstractive summarization, an interpretation is first created by analyzing the text document using machine learning or deep learning. Based on this interpretation, the machine predicts a summary. It transforms the text by paraphrasing sections of the original document.

Related Work

Initially, statistical approaches were used to compute a score for every sentence and then select the sentences with the highest scores. Several techniques were employed to calculate this score, such as TF-IDF and Bayesian models, among many. While these techniques were able to compute a sound summary by key phrase extraction, all of them were extractive approaches and were simply trimming the original text. Then the focus came onto utilizing Machine learning algorithms (and deep learning algorithms, which is the subset of Machine Learning algorithms) for summarization.

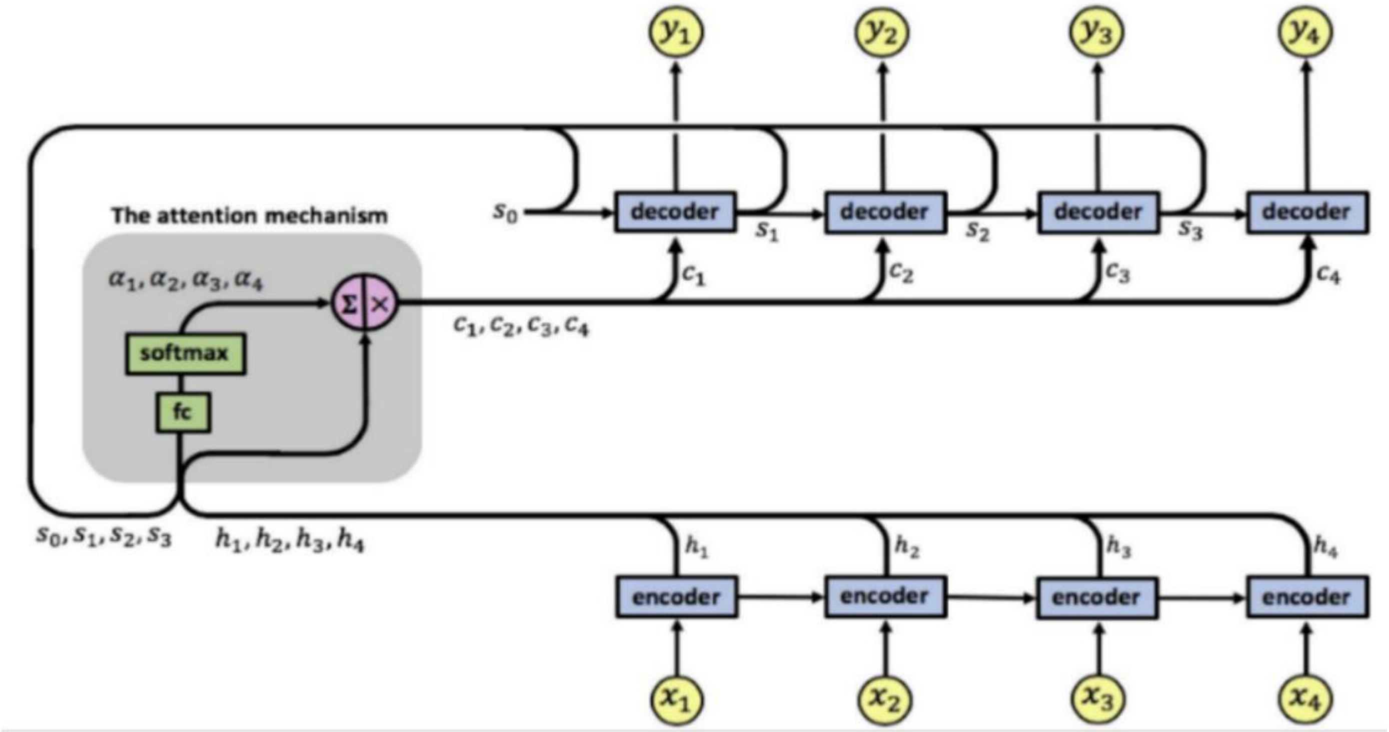


Figure 1. Attention mechanism in Encoder-Decoder architecture

Proposed Methodology

1. **Reference Link of Dataset Used**: https://www.kaggle.com/gowrishankarp/newspaper-text-summarization-cnn-dailymail
2. **Pre-processing**

The raw dataset was then cleaned using various pre-processing techniques such as:

a. **Lower Casing:** To convert the input text into the same casing format so that all capital, lower case, and mixed case are treated similarly.

b. **Eliminate Punctuation:** HTML tags and links- Removal of punctuations, links, and tags that do not add meaning to the text such as “!"#$%&\'()*+,-./: <=>?@[\\]^_{|}~`” to standardize the text.

c. **Eliminate stop words and frequently occurring words:** Removal of common words such as ‘the’, ‘a’, etc that are frequently used in a text but do not provide valuable information for downstream analysis.

d. **Stemming:** Reducing the inflected words to their root form.

e. **Lemmatization:** Reducing derived words to their base or root form while making sure that root words belong to the language.

f. **Contraction mapping:** Expanding the shortened version of words or syllables.

g. **Scaling to a range:** This is used to scale the values of the vectors, as the processing becomes faster, as it need not process large data values.

(**Linear Scaling Formula:** x_new= (x-x_ minimum)(x_maximum – x_minimum)

Log Scaling Formula: x_new=log(x_old))

h. **Feature Clipping:** If the data set contains extreme outliers, which caps all feature values above (or below) a certain value to fixed value, which may influence the training, especially in the smaller dataset.

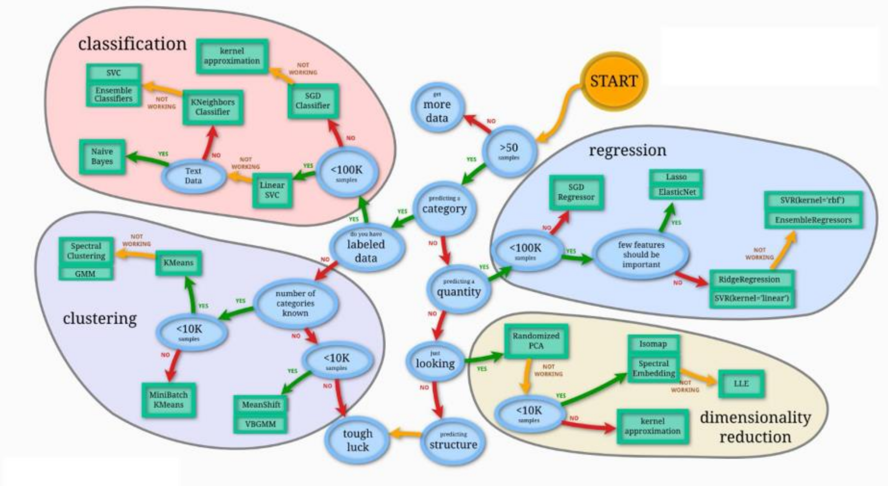


Figure 2. sci-kit-learn’s Roadmap

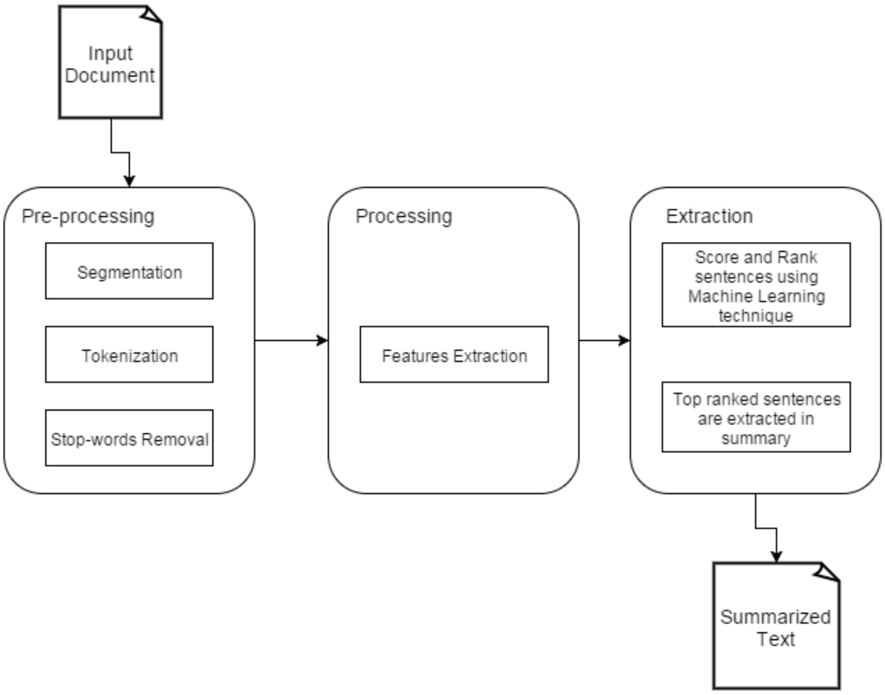


Figure 3. Workflow of the proposed methodology

Proposed Flow Chart

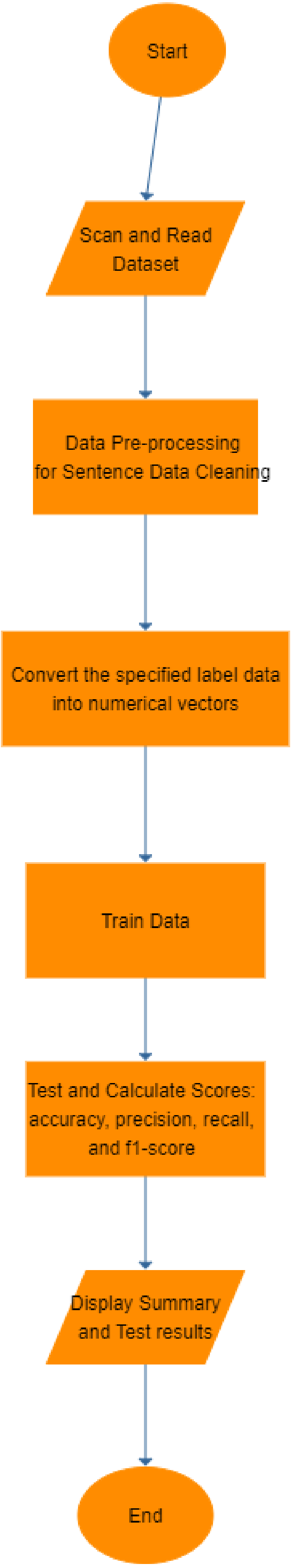

Start
Scan and Read Dataset
Data Pre-processing for Sentence Data Cleaning
Convert the specified label data into numerical vectors
Train Data
Test and Calculate Scores: accuracy, precision, recall, and f1-score
Display Summary and Test results
End

Figure 4. Flowchart of Program

Literature Survey

(With Citation)

1. **An overview of Text Summarization techniques.**

Authors: Narendra Andhale and L.A. Bewoor.

Reference Link: https://ieeexplore.ieee.org/document/7860024

Existing Work

This paper presents extractive and abstractive text summarization techniques based on statistical, machine learning, and deep learning algorithms. The abstractive summarization aims to produce a generalized summary, concisely conveying information. The extractive summarization method selects informative sentences from the document as they appear in the source based on the ranking of features of specific criteria to form a summary.

Research Gap

The results produced by this research publication lacks optimal implementation for systematic, comprehensive, and strategic testing of the text summarization techniques to aid developers directly utilizing the specific framework and technique.

2. **MOGRIFIER LSTM**

Authors: Gabor Melis, Tomas Kocisky, and Phil Blunsom.

Reference Link: https://arxiv.org/pdf/1909.01792.pdf

Existing Work

In this work, the authors propose an extension to the venerable Long Short-Term Memory in the form of mutual gating of the current input and the previous output. This mechanism affords the modeling of a richer space of interactions between inputs and their context. Equivalently, the model can be viewed as making the transition function given by the LSTM context-dependent.

Research Gap

Although the Recurrent Network (RNN) is a simple and powerful model, in practice, it has Slow and Complex training procedures with exploding gradient problems (model weights become unexpectedly large in the end). Also, the drawbacks of using the BERT pre-trained model are that it is very compute-intensive at inference time.

3. **Transformer and seq2seq model for Paraphrase Generation**

Authors: Elozino Egonmwan and Yllias Chali.

Reference Link: https://aclanthology.org/D19-5627.pdf

Existing Work

Paraphrase generation aims to improve the clarity of a sentence by using different wording that conveys a similar meaning. For better quality of generated paraphrases, the whitepaper proposes a framework that combines the effectiveness of two models, which include: transformer (using GRU-RNN) and sequence-to-sequence (seq2seq) models.

Research Gap

This also includes the problem of Slow and Complex training procedures with exploding gradient problems (model weights become unexpectedly large in the end) in the Recurrent Network (RNN) component. Along with it, In the case of long sequences, there is a high probability that the initial context has been lost by the end of the sequence. This is because, in the sequence-to-sequence model, the output sequence relies heavily on the context defined by the hidden state in the final output of the encoder, for which sequence-to-sequence is executed sequentially after RNN to resolve the associated challenge.

4. Sequence to Sequence Learning with Neural Networks

Authors: Ilya Sutskever, Oriol Vinyals, and Quoc V. Le.

Reference Link: https://arxiv.org/pdf/1409.3215.pdf

Existing Work

The author discovered that reversing the order of words in all source sentences (but not target sentences) significantly enhanced the LSTM's performance since it produced many short-term dependencies between the source and target sentences, making the optimization problem easier to solve using Deep Neural Networks (DNNs) and sequence to sequence model sequentially at the end.

Research Gap

Usually, neural networks are also more computationally expensive than traditional algorithms. Algorithms, which realize successful training of really deep neural networks, can take several weeks to train completely from scratch. It also requires much more data than traditional machine learning algorithms.

5. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.

Authors: Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio.

Reference Link: https://arxiv.org/pdf/1406.1078.pdf

Existing Work

In this paper, the authors proposed a novel neural network model called RNN Encoder-Decoder that consists of two recurrent neural networks (RNN). One RNN encodes a sequence of symbols into a fixed-length vector representation, and the other decodes the representation into another sequence of symbols. Qualitatively, it shows that the proposed model learns a semantically and syntactically meaningful representation of linguistic phrases. It also uses Word Penalization (WP), Continuous-Space Language Model (CSLM), and Baseline Configuration with recurrent neural networks.

Research Gap

They suffer from some important drawbacks, including a very long training time and limitations on the number of context words. But recently, there have been many extensions to language models designed just to address these problems and require much more data than other algorithms discussed.

6. Review of automatic text summarization techniques and methods

Authors: Adhika Pramita Widyassari, Supriadi Rustad, Guruh Fajar Shidik, Edi Noersasongko, Abdul Syukur, Affandy Affandy, and De Rosal Ignatius Moses Setiadi

Reference Link: https://www.sciencedirect.com/science/article/pii/S1319157820303712?via%3Dihub

Existing Work

The results of the analysis provide an in-depth explanation of the results that are the focus of their publication in the field of text summarization. It also describes the techniques and methods that are often used by researchers as a comparison and means for developing methods. At the end of this paper, several recommendations for opportunities and challenges related to text summarization research are mentioned.

Research Gap

The important thing that is considered interesting from the review that has been done is the results of the analysis which states that implementation of extractive summaries is relatively easier than abstractive summaries. This may result also in semantics loss in extractive methods like the TF-IDF technique. Also, real-time optimization through parallelism of models and frameworks cannot be done, as in the case of neural networks.

7. A Review Paper on Text Summarization

Authors: Deepali K. Gaikwad and C. Namrata Mahender

Reference Link: https://www.ijarcce.com/upload/2016/march-16/IJARCCE%2040.pdf

Existing work

The two major approaches i.e., extractive and abstractive summarization is discussed in detail. The technique deployed for summarization ranges from structured to unstructured linguistics. In Indian many languages (Hindi, Punjabi, Bengali, Kannada, Malayalam, Telugu, and Tamil), the work has been done, but presently they are in infancy state and of extractive methods. This paper provides an abstract view of the present scenario of research work for text summarization.

Research Gap

As abstractive summarization requires more learning and a period to get trained. The abstractive approach is also a bit complex to understand and implement than the extractive approach but provides a more meaningful and appropriate summary. Through the study, it is also observed that very less work is done using abstractive methods on Indian languages. Also, the results produced by this research publication lacks optimal implementation for systematic, comprehensive, and strategic testing of the text summarization techniques.

8. Text Summarization Techniques: A Brief Survey

Authors: Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut

Reference Link: https://arxiv.org/abs/1707.02268

Existing work

In this review, the main approaches to automatic text summarization are described. The publication reviews the different processes for summarization and describes the effectiveness and shortcomings of the different methods. The testing of techniques and methods in this review used datasets of text with large volume is an invaluable source of information and knowledge, which needs to be effectively removed and then summarized.

Research Gap

In this paper, the authors emphasized various extractive approaches for single and multi-document summarization. For this, they didn’t review or propose any technique or process method to merge documents or results from various sources. Although it is not feasible to explain all diverse algorithms and approaches comprehensively in the paper, it does not provide a comprehensive explanation of various approaches through technical diagrams and inadequate details.

9. Natural Language Processing (NLP) based Text Summarization

Authors: Ishitva Awasthi, Kuntal Gupta, Prabjot Singh Bhogal, Sahejpreet Singh Anand, and Piyush Kumar Soni

Reference Link: https://ieeexplore.ieee.org/document/9358703

Existing Work

Based on linguistic and statistical characteristics, the implications of sentences are calculated. A study of extractive and abstract methods of NLP for summarizing texts has been made in this paper. This paper also analyses methods that yield a less repetitive and more concentrated summary. This also studies various other techniques and results using Intrinsic Evaluation and Extrinsic Evaluation of Automated Text Summarization.

Research Gap

The below problem is also brought out loopholes based on the following problems below.

a. Anaphora Problem: To understand which pronoun used in the article is a substitution for which of the previously introduced terms.
b. Cataphora Problem: To understand which ambiguous words or explanations are used to refer to a particular term before even introducing the term itself.

Furthermore, there are various techniques and methods, researchers specializing in Natural Language Processing (NLP) are particularly drawn to extractive methods (they are fast but are more susceptible to semantics loss).

10. A survey automatic text summarization

Authors: Oguzhan Tas and Farzad Kiyani

Reference Link: https://www.pressacademia.org/archives/pap/v5/29.pdf

Existing Work

There are two different groups of text summarization, Inductive and Informative. Indicative summarization gives the main idea of the text to the user in around 5 percent of a given text. The informative summarization system gives brief information of the main text in around 20 percent of the given text. Furthermore, summarization methods can be classified according to the source which can be single or multiple document summarization using extractive and abstractive techniques. So, the real aim is not only to remove redundancy and identify correct text for a summary but also to provide novelty and ensure that the final summary should be coherent and complete in itself.

Research Gap

The abstractive and extractive summary techniques used for parsing and describing the content of the text require heavy computation (in terms of memory and period) from natural language processing, including for generating, cleaning, and analyzing grammars, punctuations, lexemes, and tokens. So, this paper is focussing on fast extractive summarization methods, but are susceptible to semantic loss.

11. A Survey of Extractive and Abstractive Text Summarization Techniques

Authors: Vipul Dalal and Latest Malik

Reference Link: https://ieeexplore.ieee.org/abstract/document/6754792

Existing Work

As there exists an urgent need for the discovery of knowledge embedded in digital documents. This paper intends to investigate techniques and methods used by researchers for automatic text summarization. Special attention is paid to Bio-inspired methods for text summarization. Here, the equation: summarization = topic identification + interpretation +generation, has been proposed.

Research Gap

Abstractive summarization approaches outperform extractive approaches but are more expensive computationally. Combining bio-inspired techniques with abstractive approach should optimize the computation cost in terms of computational complexity but are difficult to implement. Also, the proposal, implementation, and testing of a novel technique based on a bio-inspired abstractive approach for automatic text summarization have not been done.

12. Text summarization from legal documents: a survey

Authors: Ambedkar Kanapala, Sukomal Pal, and Rajendra Pamula

Reference Link: https://link.springer.com/article/10.1007/s10462-017-9566-2

Existing Work

The authors discuss different datasets and metrics used in summarization and compare performances of different approaches, first in general and then focused on legal text. The review also mentions highlights of different summarization techniques. It briefly covers a few software tools used in legal text summarization. We finally conclude with some future research directions.

Research Gap

There is a lack of a multi-document summarization approach, to get a single result from similar cases, which can provide the legal practitioners a brief and holistic view of a particular type of court-cases. There exist several concerns, with the infrequent use of specific word tokens, which can result in the elimination of crucial phrases or the acceptance of unneeded sentences containing these tokens. Thus, resulting in an unnecessary large summary or semantics loss.

13. A survey of automatic text summarization techniques for Indian and foreign languages

Authors: Prachi Shah and Nikita P. Desai

Reference Link: https://ieeexplore.ieee.org/abstract/document/7755587

Existing Work

This paper presents several text summarization techniques and provides a survey of text summarization approaches for various Indian (Tamil, Kannada, Bengali, Bangala, Punjabi, and Hindi) and foreign languages (English, Chinese, Arabic, Turkish, and Swedish). It has also described a few challenges which are still under research.

Research Gap

We can also conclude that different combination of features works differently for different types of content, which need to be selected and encoded into the supervised learning's application. Hence, it is challenging to create a single summarizer for different types of content. It also aims to utilize more features for extracting sentences in non-primary languages. Also, it lacks some machine learning techniques (especially clustering algorithms for unsupervised learning) for comparison.

14. Neural Networks for Joint Sentence Classification in Medical Paper Abstracts

Authors: Franck Dernoncourt, Ji Young Lee, and Peter Szolovits

Reference Link: https://arxiv.org/abs/1612.05251

Existing Work

Existing models based on artificial neural networks (ANNs) for sentence classification often do not incorporate the context in which sentences appear and classify sentences individually. In this work, the authors presented an ANN architecture that combines the effectiveness of typical ANN models to classify sentences in isolation, with the strength of structured prediction.

Research Gap

In this article, we have presented an ANN architecture to classify sentences that appear in specified sequence using joint learning (for both extractive and abstractive hybrid models). It improves the quality of the predictions, but has the problem of high computational complexity, especially when the approaches like Support Vector Machines (SVMs) have been used for the training.

15. Improving Performance of Text Summarization

Authors: S.A.Babara and Pallavi D.Patil

Reference Link: https://www.sciencedirect.com/science/article/pii/S1877050915000952

Existing Work

The summary of the document is created based upon the level of the importance of the sentences in the document. This paper focuses on the Fuzzy logic Extraction approach for text summarization and the semantic approach of text summarization using Latent Semantic Analysis. This focuses on high relevance (rank) from the document based on semantics, word, and sentence tokens features. The proposed method improves the quality of summary by incorporating the latent semantic analysis into the sentence feature extracted fuzzy logic system to extract the semantic relations between concepts in the original text.

Research Gap

The focus of this paper is narrow: summarization of documents, but the ideas are more broadly applicable, including transcripts. It also needs to extend the proposed method for multi-document summarization with large data sets and domain-specific data.

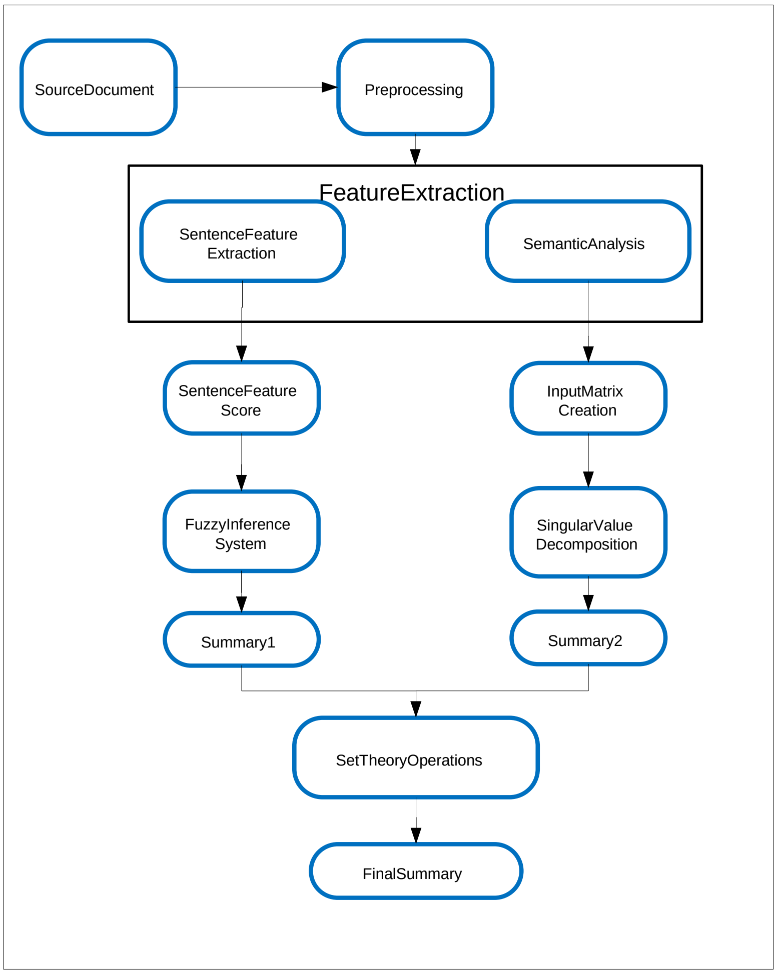


Figure 5. Proposed Architecture

16. A review on Text summarization Techniques

Authors: Pradeepika Verma and Anshul Verma

Reference Link: https://www.bhu.ac.in/research_pub/jsr/Volumes/JSR_64_01_2020/48.pdf

Existing work

This requires a precise analysis of the text in various steps such as semantic analysis, lexical relations, named entity recognition, etc., which can be accomplished with a great deal of word knowledge only. In particular, the system requires to identify the most relevant/significant contents of the text, extract them, order them, and return them to the user. Although the extractive summarization task has been a popular research topic since 1958 (Luhn, 1958), yet it is a great challenge to summarize a text automatically using a computational system like a human-generated summary.

Research gap

A summary is more informative as much as it contains non-redundant content. Most of the existing approaches focus on finding relevant content from the document(s) and extracting them to generate the summary. Thus, considering all possible text features for the assessment of the sentences increases complexity as well as irrelevancy. Moreover, in a reference summary, all the considered features do not manifest in the same ratio.

Therefore, if we consider all the features in the same ratio and assess the sentences accordingly, then this hypothesis may create irrelevant content in the generated summary. Hence, it is also required to know the proper ratio from the given dataset in which they should be presented in the summary. C. Problem of loss of coverage. Coverage of topics of a document in the summary is an important aspect of generic text summarization.

Hence, they fail to produce a good summary in the case of generic summarization. This problem arises mainly in the case of multi-document summarization where the number of topics in documents is much higher than in a single document. In the literature of text summarization, there exist some approaches which focus on maximizing the coverage while minimizing the redundancy. Suppose they get the best result at a point where the redundancy is minimum, then there are very high chances of loss of coverage.

D. Problem of non-readability and less cohesive content. A good summary should be readable and cohesive. This paper presents a summarization method that takes into account readability and cohesion parameters to generate the summaries of the document.

17. Automatic text summarization and its methods - a review

Author: Neelima Bhatia and Arunima Jaiswal

Reference Link: https://ieeexplore.ieee.org/document/7508049

Existing work

Text summarization has grown so uses such as Due to the enormous aggregate of information getting augmented on the internet; it is challenging for the user to verve through altogether the information accessible on the web. The large availability of internet content partakes constrained a broad research area in the extent of automatic text summarization contained by the Natural Language Processing (NLP), especially statistical machine learning communal.

Research Gap

Automatic text summarization is an old challenge but the current research direction diverts towards emerging trends in biomedicine, product review, education domains, emails, and blogs. The purpose of extractive document summarization is to automatically select several indicative sentences, passages, or paragraphs from the original document. Text summarization approaches based on Neural Network, Graph-Theoretic, Fuzzy, and Cluster have, to an extent, succeeded in making an effective summary of a document. Both extractive and abstractive methods have been researched.

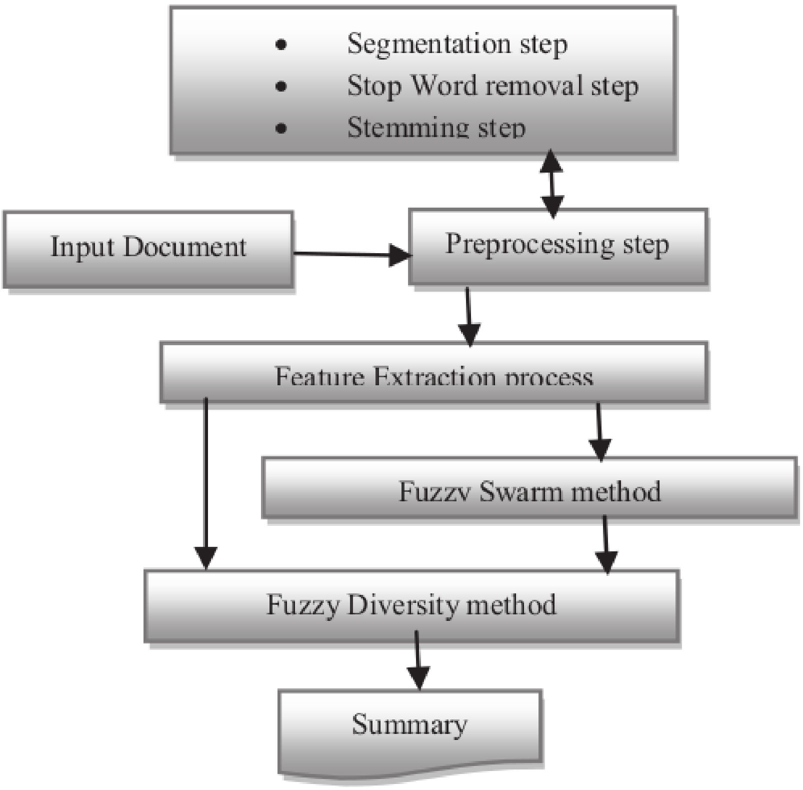


Figure 6. The output of the fuzzy logic controller.

(The resolution of output is done by a procedure known as defuzzification.)

Implementation

We will use Abstractive summarization The difference between Abstractive and extractive. Extraction is just based on the Tf-IDF method to find the relation between the word and extract it. So, this will neglect the schematic of the document whereas the Abstractive method understands the language and acts relatively.

For Example:

Sentence: -

"Alice and Bob took the train to visit the zoo. They saw a baby giraffe, a lion, and a flock of colorful tropical birds."

Abstractive: -

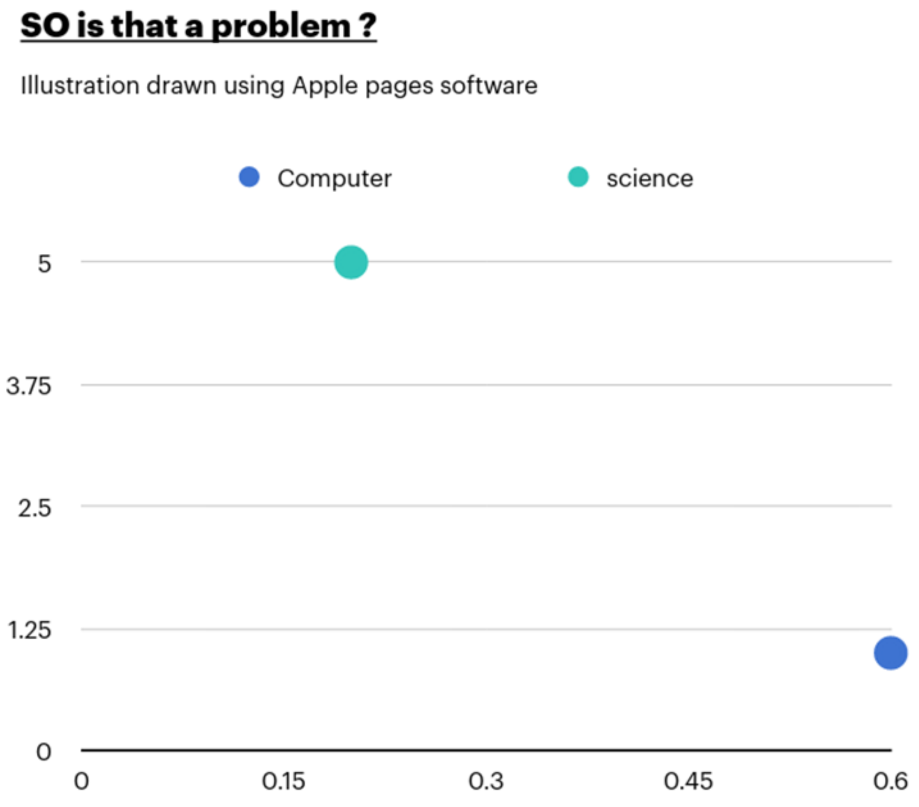
"Alice and Bob visited the zoo and saw animals and birds".

Extractive: -

"Alice and Bob visit the zoo. saw a flock of birds."

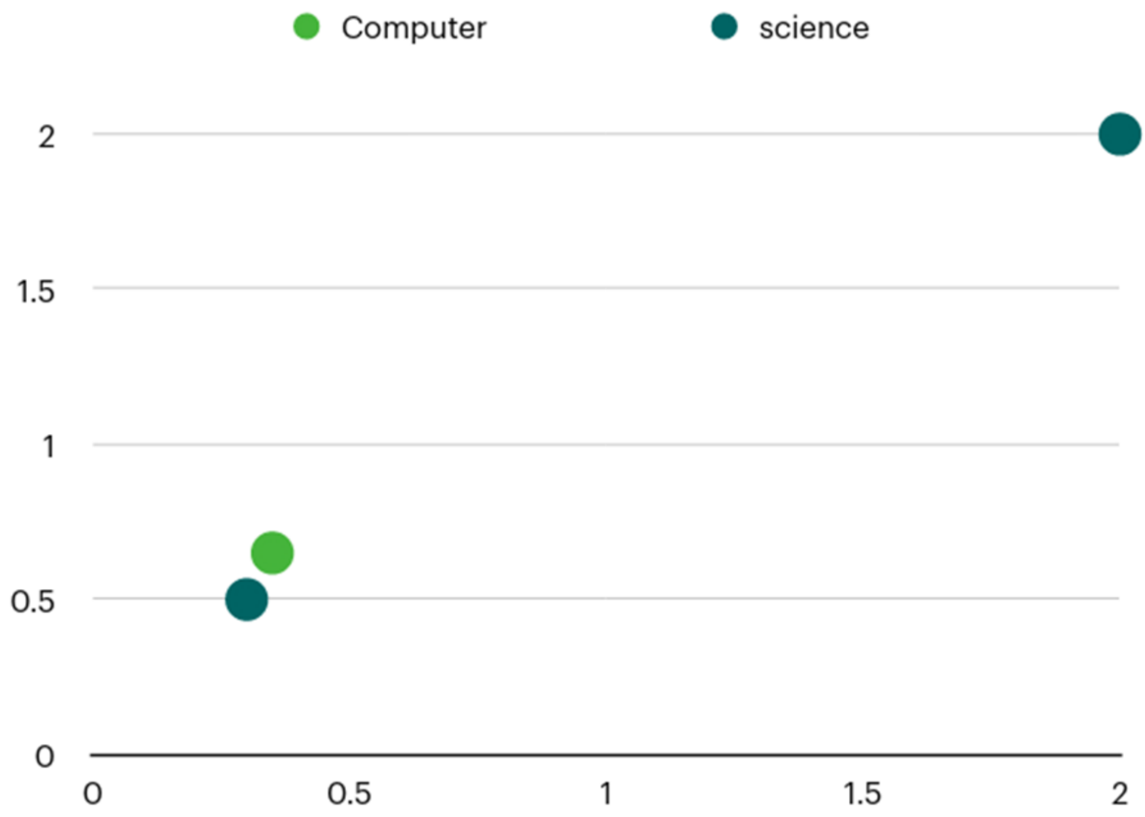
Reference: https://ai.googleblog.com/2016/08/text-summarization-with-tensorflow.html

We have tested some major online pre-trained models but what they lack is domain-based segmentation. We have found out that there is around half a million word in according to the standard Oxford dictionary. But when it comes to the medical field there are around 3 million words. How is that possible? That is because words are relative to the domain. Suppose assume the name "Pulmonary cardiac disease "These are the 3 separate words in English but with respective medical science this is a single word so our model has to understand it.



(The above graph Is drawn just illustrative purpose)

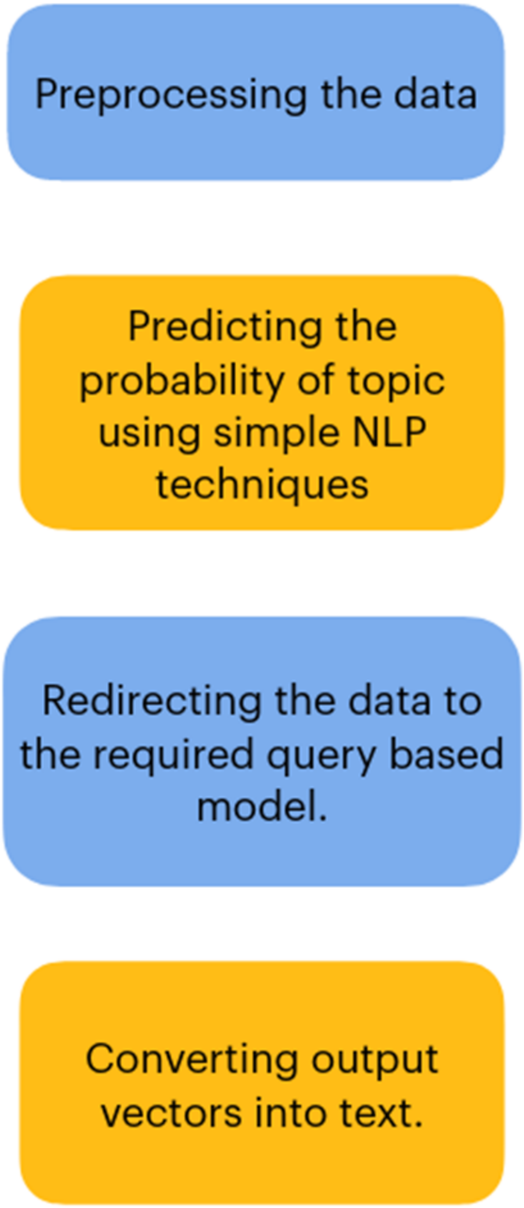
The above graph is illustrative of training a simple LSTM or transformer model of different research papers or articles of different domains to the single model. We can see the model variation for 'Computer' and 'science' as 2 different words. This lets us train the model with only computer science papers and articles. So, we will be using sequence models like Last, GRU, etc so the model may recognize the distance between the computer and science as a sequence.



Encoding the Data

The main problem with the word is with encoding or word into some unique vectors in the space so that the output is also a unique vector in space and is also a word there are some techniques and pre-build models such as GloVE, word2vec models but we will be choosing word embeddings for training the model.

WORKFLOW OF THE PROJECT:-


Preprocessing the data
Predicting the probability of topic using simple NLP techniques
Redirecting the data to the required query based model.
Converting output vectors into text.

Hardware and Tools Used

a) Google Colab
b) Tensorflow
c) Numpy
d) Pandas
e) Sklearn
f) Seaborn (Matplotlib)

Datasets

We tried to experiment with our Models on 2 datasets as follows below.

a) Weather Report (Small Dataset)
b) Medial Report Summary (Large Dataset)

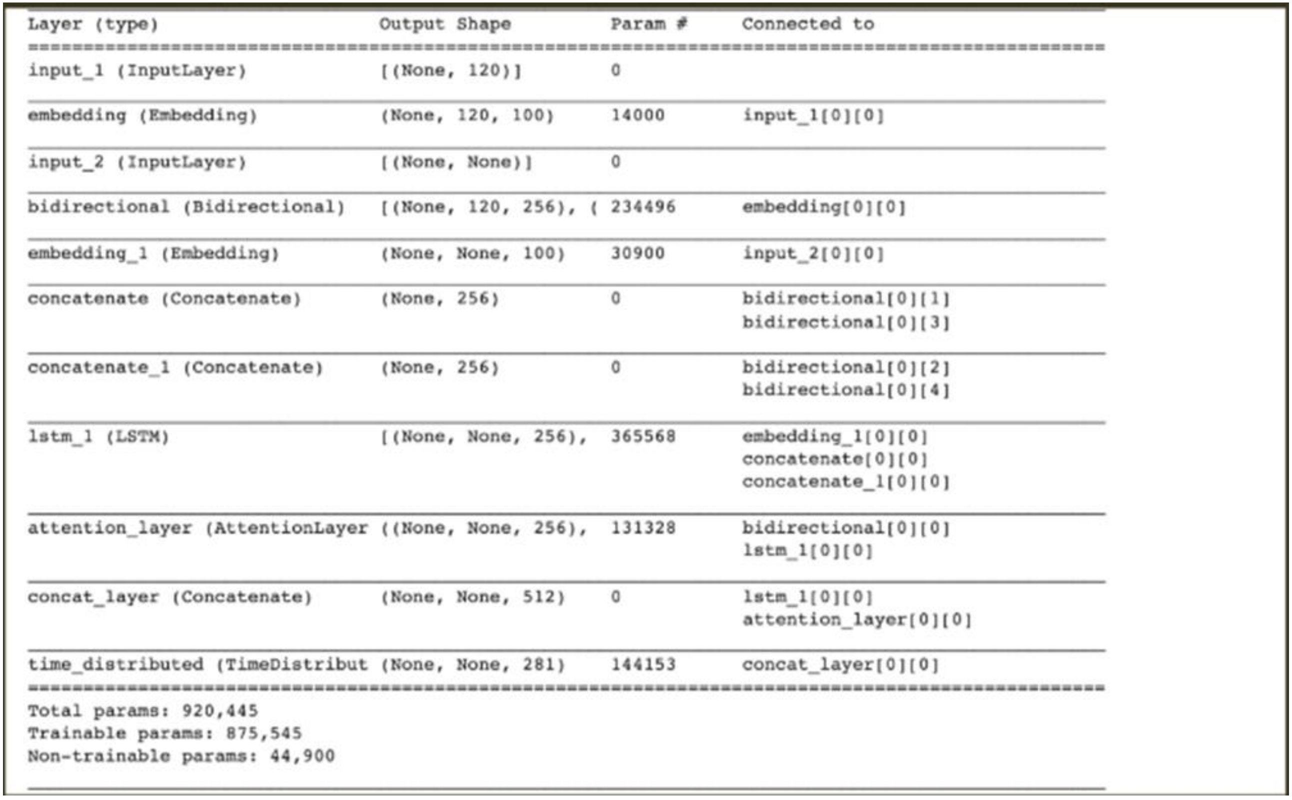
Reference Dataset: https://arxiv.org/abs/1710.06071

Table for Model Names and Dataset Used

Dataset	Model	Model's Number
Weather report	Baseline Naïve Bayers	Model_0_1
Weather_report	Dense Neural Network	Model_1_1
Health Report	Navie Bayers	Model_0
Health Report	Conv1D with token embeddings	Model_1
Health Report	Feature extraction with pretrained token embeddings	Model_2
Health Report	Conv1D with character embeddings	Model_3
Health Report	Combining pretrained token embeddings + character embeddings	Model_4
Health Report	Transfer Learning with pretrained token embeddings + character embeddings + positional embeddings	Model_5

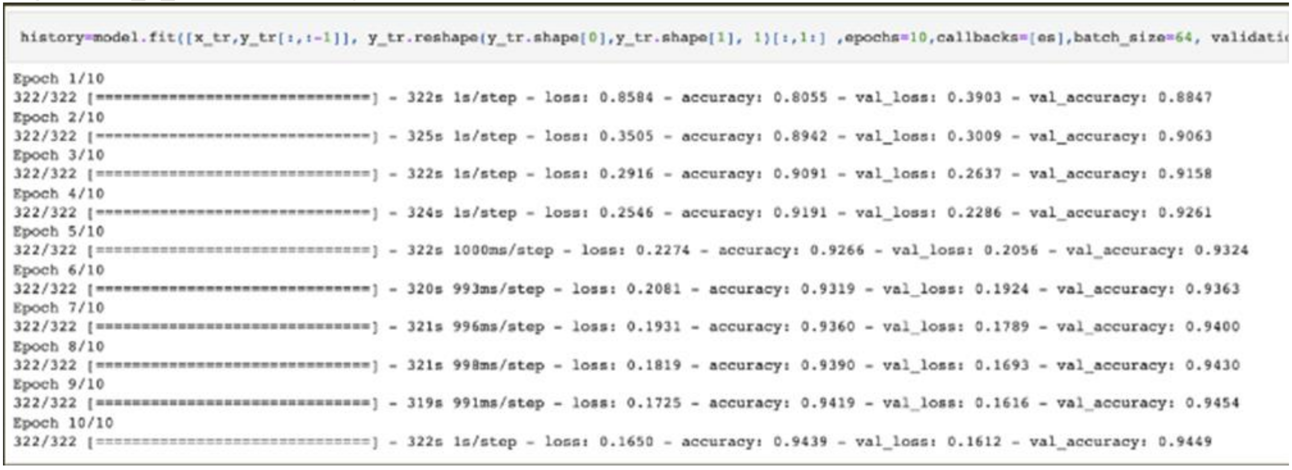
Results

The architecture of Model_1_1



```
Layer (type)                    Output Shape          Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 120)]         0

embedding (Embedding)           (None, 120, 100)      14000       input_1[0][0]

input_2 (InputLayer)            [(None, None)]        0

bidirectional (Bidirectional)   [(None, 120, 256), (  234496      embedding[0][0]

embedding_1 (Embedding)         (None, None, 100)     30900       input_2[0][0]

concatenate (Concatenate)       (None, 256)           0           bidirectional[0][1]
                                                                  bidirectional[0][3]

concatenate_1 (Concatenate)     (None, 256)           0           bidirectional[0][2]
                                                                  bidirectional[0][4]

lstm_1 (LSTM)                   [(None, None, 256),   365568      embedding_1[0][0]
                                                                  concatenate[0][0]
                                                                  concatenate_1[0][0]

attention_layer (AttentionLayer ((None, None, 256),   131328      bidirectional[0][0]
                                                                  lstm_1[0][0]

concat_layer (Concatenate)      (None, None, 512)     0           lstm_1[0][0]
                                                                  attention_layer[0][0]

time_distributed (TimeDistribut (None, None, 281)     144153      concat_layer[0][0]
==================================================================================================
Total params: 920,445
Trainable params: 875,545
Non-trainable params: 44,900
```

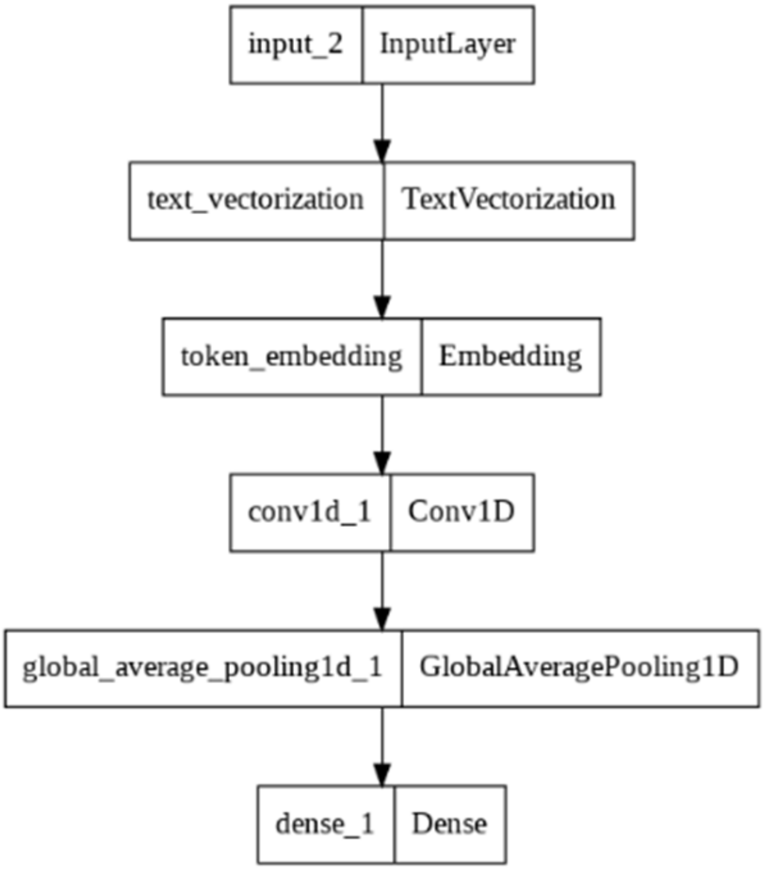
Training Model_1_1 with accuracy and loss



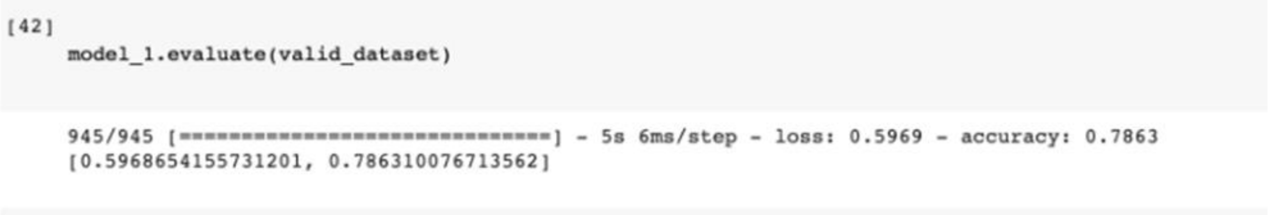
```
history=model.fit([x_tr,y_tr[:,:-1]], y_tr.reshape(y_tr.shape[0],y_tr.shape[1], 1)[:,1:] ,epochs=10,callbacks=[es],batch_size=64, validati
```

```
Epoch 1/10
322/322 [==============================] - 322s 1s/step - loss: 0.8584 - accuracy: 0.8055 - val_loss: 0.3903 - val_accuracy: 0.8847
Epoch 2/10
322/322 [==============================] - 325s 1s/step - loss: 0.3505 - accuracy: 0.8942 - val_loss: 0.3009 - val_accuracy: 0.9063
Epoch 3/10
322/322 [==============================] - 322s 1s/step - loss: 0.2916 - accuracy: 0.9091 - val_loss: 0.2637 - val_accuracy: 0.9158
Epoch 4/10
322/322 [==============================] - 324s 1s/step - loss: 0.2546 - accuracy: 0.9191 - val_loss: 0.2286 - val_accuracy: 0.9261
Epoch 5/10
322/322 [==============================] - 322s 1000ms/step - loss: 0.2274 - accuracy: 0.9266 - val_loss: 0.2056 - val_accuracy: 0.9324
Epoch 6/10
322/322 [==============================] - 320s 993ms/step - loss: 0.2081 - accuracy: 0.9319 - val_loss: 0.1924 - val_accuracy: 0.9363
Epoch 7/10
322/322 [==============================] - 321s 996ms/step - loss: 0.1931 - accuracy: 0.9360 - val_loss: 0.1789 - val_accuracy: 0.9400
Epoch 8/10
322/322 [==============================] - 321s 998ms/step - loss: 0.1819 - accuracy: 0.9390 - val_loss: 0.1693 - val_accuracy: 0.9430
Epoch 9/10
322/322 [==============================] - 319s 991ms/step - loss: 0.1725 - accuracy: 0.9419 - val_loss: 0.1616 - val_accuracy: 0.9454
Epoch 10/10
322/322 [==============================] - 322s 1s/step - loss: 0.1650 - accuracy: 0.9439 - val_loss: 0.1612 - val_accuracy: 0.9449
```

Architecture of Model_1

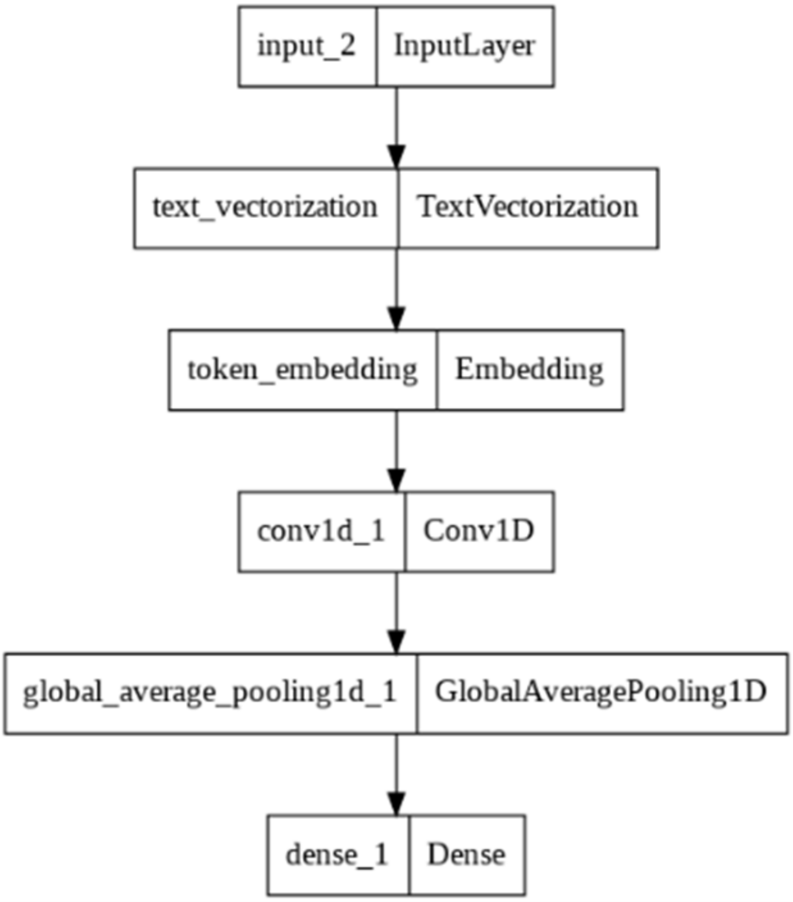

input_2 | InputLayer
text_vectorization | TextVectorization
token_embedding | Embedding
conv1d_1 | Conv1D
global_average_pooling1d_1 | GlobalAveragePooling1D
dense_1 | Dense

Training Model_1 With accuracy and Loss

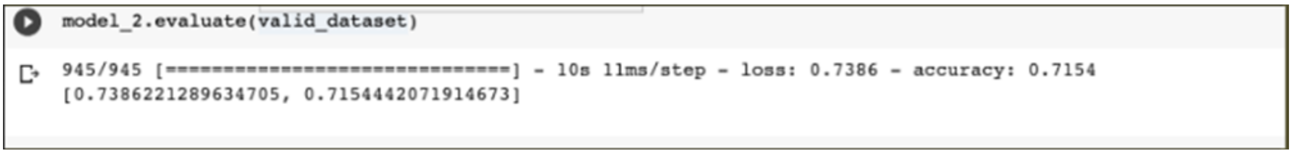


```
[42]
    model_1.evaluate(valid_dataset)

    945/945 [==============================] - 5s 6ms/step - loss: 0.5969 - accuracy: 0.7863
    [0.5968654155731201, 0.786310076713562]
```

Architecture of Model_2

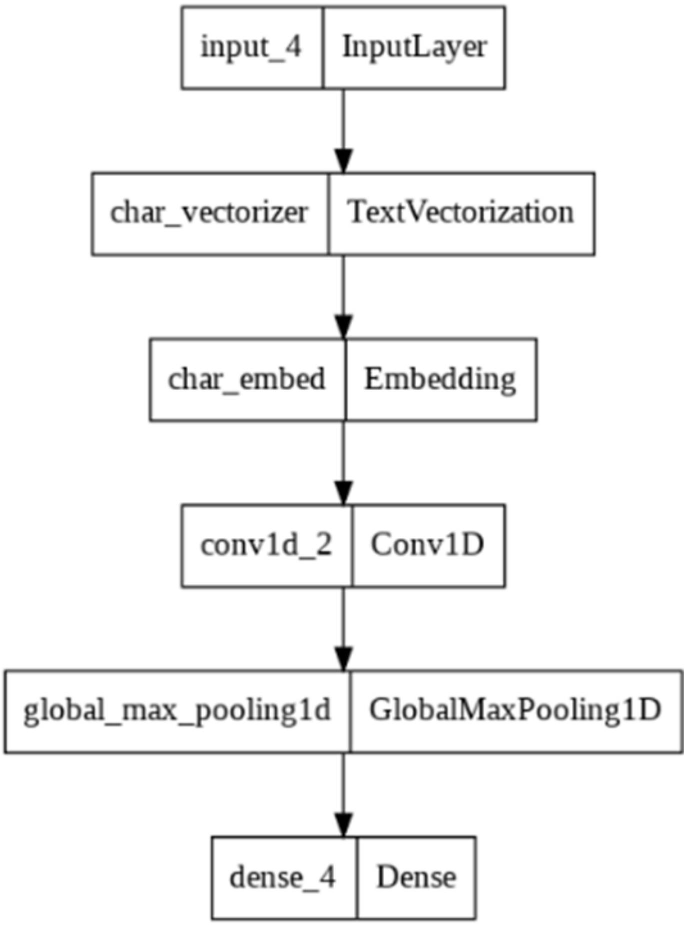

input_2 | InputLayer
text_vectorization | TextVectorization
token_embedding | Embedding
conv1d_1 | Conv1D
global_average_pooling1d_1 | GlobalAveragePooling1D
dense_1 | Dense

Training Model_2 with Accuracy and Score

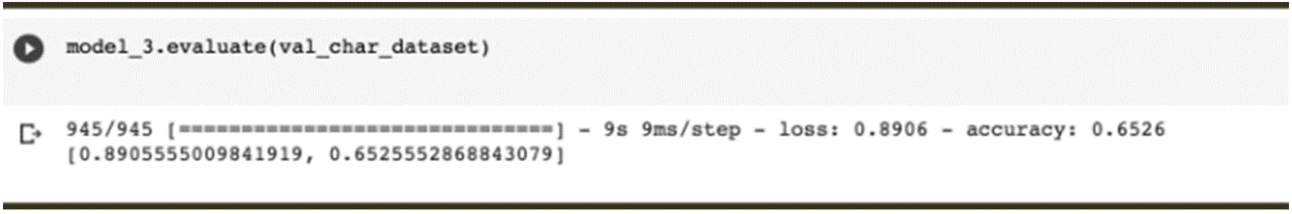


```
model_2.evaluate(valid_dataset)

945/945 [==============================] - 10s 11ms/step - loss: 0.7386 - accuracy: 0.7154
[0.7386221289634705, 0.7154442071914673]
```

The architecture of Model_3 (Cov1D with character Embeddings)



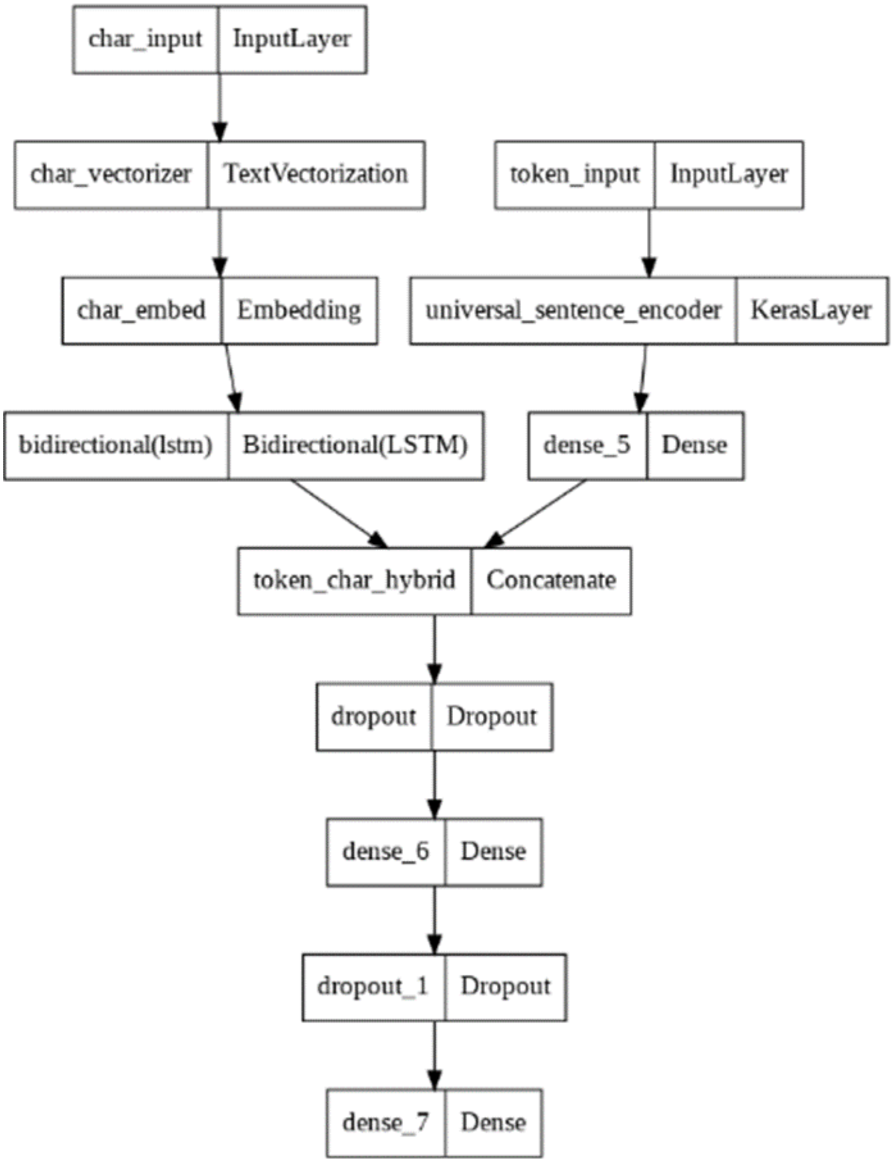
Training Model_3 with accuracy and Score



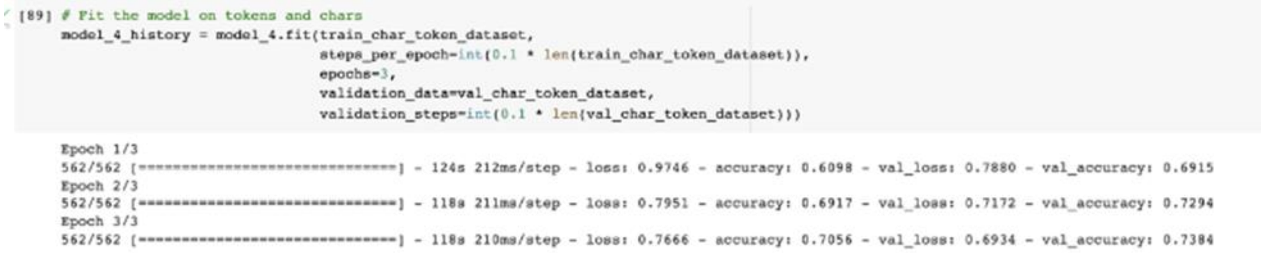
```
model_3.evaluate(val_char_dataset)
```

```
945/945 [==============================] - 9s 9ms/step - loss: 0.8906 - accuracy: 0.6526
[0.8905555009841919, 0.6525552868843079]
```

The architecture of Model_4 (pre-trained token embeddings and Character Embeddings)



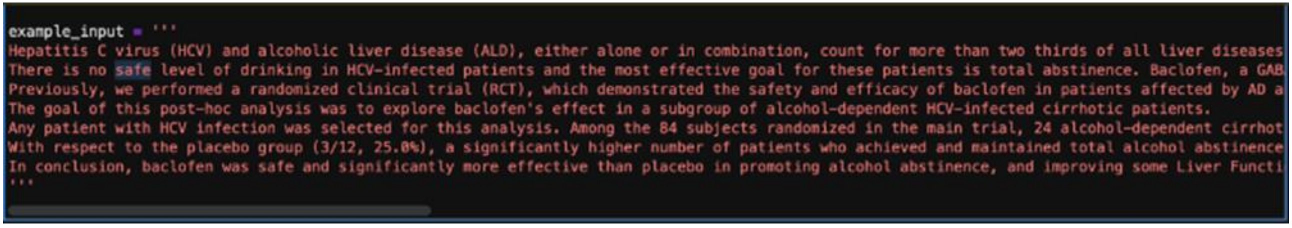
Training Model_4 with Accuracy and Score



```
[89] # Fit the model on tokens and chars
     model_4_history = model_4.fit(train_char_token_dataset,
                                   steps_per_epoch=int(0.1 * len(train_char_token_dataset)),
                                   epochs=3,
                                   validation_data=val_char_token_dataset,
                                   validation_steps=int(0.1 * len(val_char_token_dataset)))

     Epoch 1/3
     562/562 [==============================] - 124s 212ms/step - loss: 0.9746 - accuracy: 0.6098 - val_loss: 0.7880 - val_accuracy: 0.6915
     Epoch 2/3
     562/562 [==============================] - 118s 211ms/step - loss: 0.7951 - accuracy: 0.6917 - val_loss: 0.7172 - val_accuracy: 0.7294
     Epoch 3/3
     562/562 [==============================] - 118s 210ms/step - loss: 0.7666 - accuracy: 0.7056 - val_loss: 0.6934 - val_accuracy: 0.7384
```

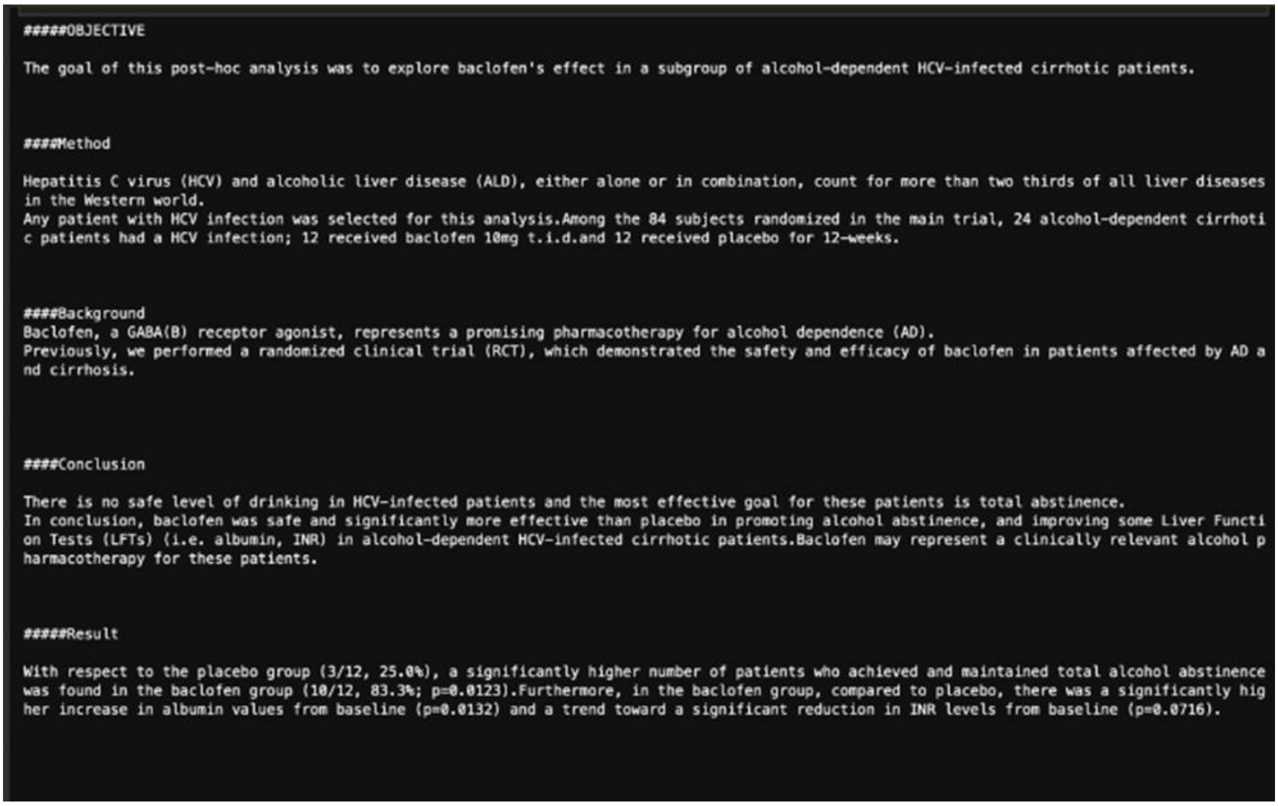
So, in the Above model_4, we can see that the One trained Model embedded tokens are taken as one Input and another Character Embedded is used as another one. This Helps the model to Differentiate between the words and characters.

Testing Model 4
Example Input



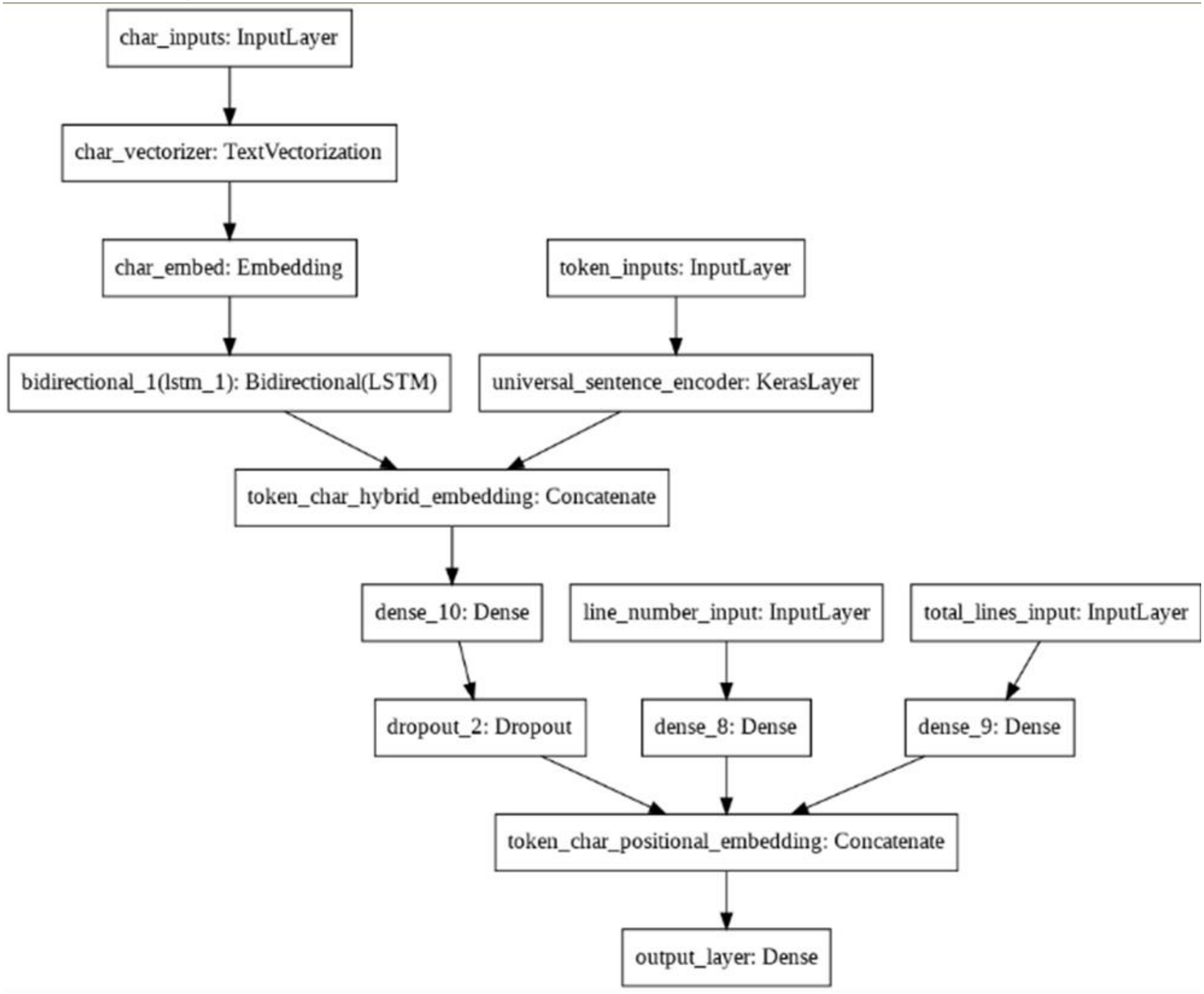
```
example_input = '''
Hepatitis C virus (HCV) and alcoholic liver disease (ALD), either alone or in combination, count for more than two thirds of all liver diseases
There is no safe level of drinking in HCV-infected patients and the most effective goal for these patients is total abstinence. Baclofen, a GAB
Previously, we performed a randomized clinical trial (RCT), which demonstrated the safety and efficacy of baclofen in patients affected by AD a
The goal of this post-hoc analysis was to explore baclofen's effect in a subgroup of alcohol-dependent HCV-infected cirrhotic patients.
Any patient with HCV infection was selected for this analysis. Among the 84 subjects randomized in the main trial, 24 alcohol-dependent cirrhot
With respect to the placebo group (3/12, 25.0%), a significantly higher number of patients who achieved and maintained total alcohol abstinence
In conclusion, baclofen was safe and significantly more effective than placebo in promoting alcohol abstinence, and improving some Liver Functi
'''
```

Output

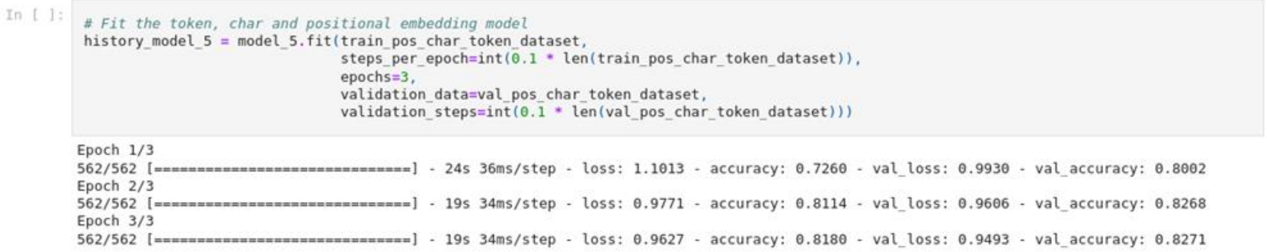


```
#####OBJECTIVE

The goal of this post-hoc analysis was to explore baclofen's effect in a subgroup of alcohol-dependent HCV-infected cirrhotic patients.



####Method

Hepatitis C virus (HCV) and alcoholic liver disease (ALD), either alone or in combination, count for more than two thirds of all liver diseases
in the Western world.
Any patient with HCV infection was selected for this analysis.Among the 84 subjects randomized in the main trial, 24 alcohol-dependent cirrhoti
c patients had a HCV infection; 12 received baclofen 10mg t.i.d.and 12 received placebo for 12-weeks.



####Background
Baclofen, a GABA(B) receptor agonist, represents a promising pharmacotherapy for alcohol dependence (AD).
Previously, we performed a randomized clinical trial (RCT), which demonstrated the safety and efficacy of baclofen in patients affected by AD a
nd cirrhosis.



####Conclusion

There is no safe level of drinking in HCV-infected patients and the most effective goal for these patients is total abstinence.
In conclusion, baclofen was safe and significantly more effective than placebo in promoting alcohol abstinence, and improving some Liver Functi
on Tests (LFTs) (i.e. albumin, INR) in alcohol-dependent HCV-infected cirrhotic patients.Baclofen may represent a clinically relevant alcohol p
harmacotherapy for these patients.



#####Result

With respect to the placebo group (3/12, 25.0%), a significantly higher number of patients who achieved and maintained total alcohol abstinence
was found in the baclofen group (10/12, 83.3%; p=0.0123).Furthermore, in the baclofen group, compared to placebo, there was a significantly hig
her increase in albumin values from baseline (p=0.0132) and a trend toward a significant reduction in INR levels from baseline (p=0.0716).
```

The architecture of Model-5 Transfer Learning with pre-trained token embeddings + character embeddings + positional embeddings

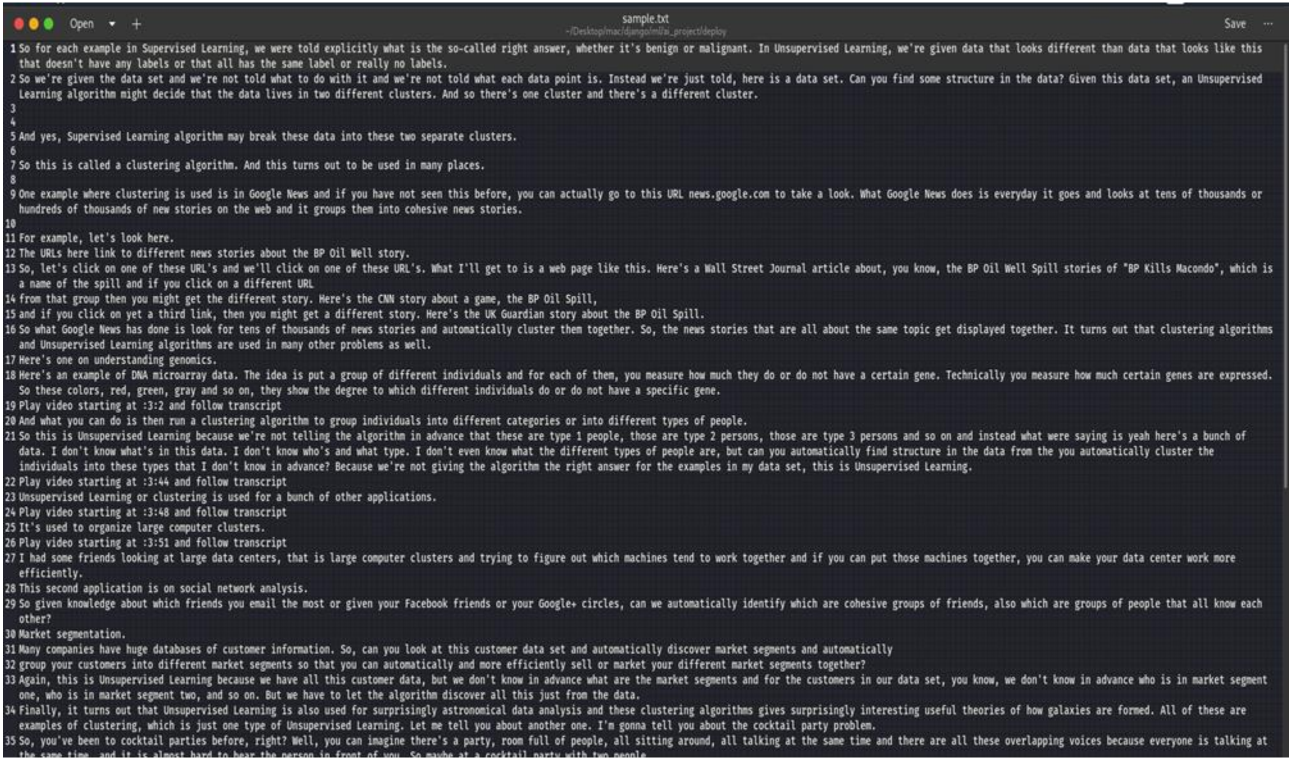

char_inputs: InputLayer
char_vectorizer: TextVectorization
char_embed: Embedding
token_inputs: InputLayer
bidirectional_1(lstm_1): Bidirectional(LSTM)
universal_sentence_encoder: KerasLayer
token_char_hybrid_embedding: Concatenate
dense_10: Dense
line_number_input: InputLayer
total_lines_input: InputLayer
dropout_2: Dropout
dense_8: Dense
dense_9: Dense
token_char_positional_embedding: Concatenate
output_layer: Dense

Model_5 with accuracy and Score



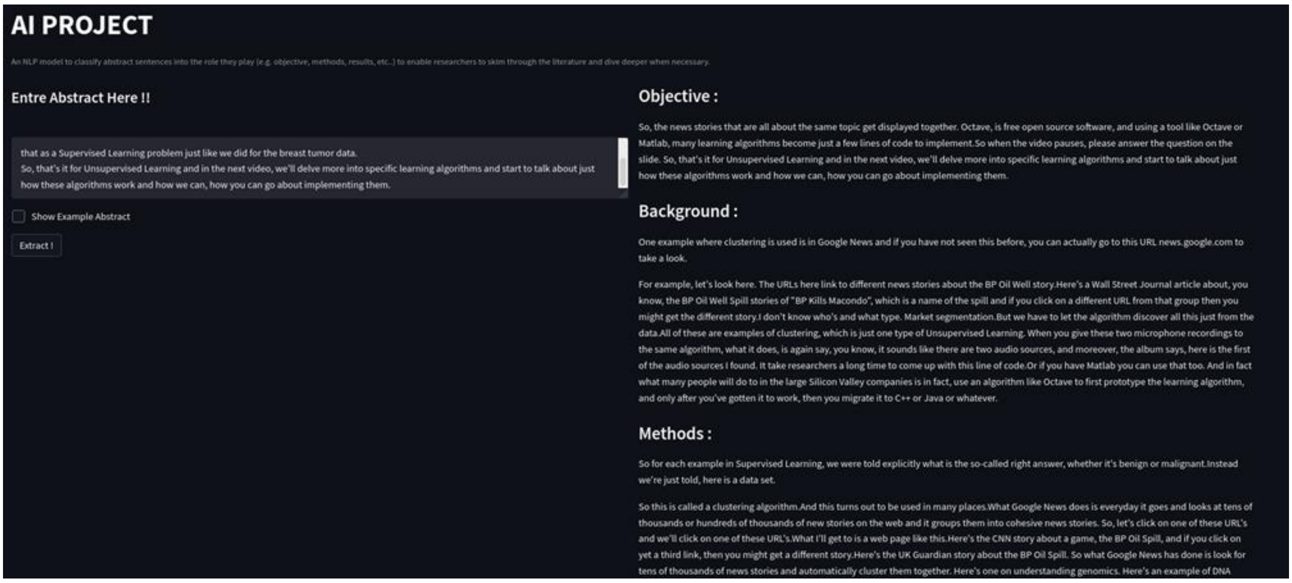
```
In [ ]:  # Fit the token, char and positional embedding model
         history_model_5 = model_5.fit(train_pos_char_token_dataset,
                                       steps_per_epoch=int(0.1 * len(train_pos_char_token_dataset)),
                                       epochs=3,
                                       validation_data=val_pos_char_token_dataset,
                                       validation_steps=int(0.1 * len(val_pos_char_token_dataset)))

         Epoch 1/3
         562/562 [==============================] - 24s 36ms/step - loss: 1.1013 - accuracy: 0.7260 - val_loss: 0.9930 - val_accuracy: 0.8002
         Epoch 2/3
         562/562 [==============================] - 19s 34ms/step - loss: 0.9771 - accuracy: 0.8114 - val_loss: 0.9606 - val_accuracy: 0.8268
         Epoch 3/3
         562/562 [==============================] - 19s 34ms/step - loss: 0.9627 - accuracy: 0.8180 - val_loss: 0.9493 - val_accuracy: 0.8271
```

Testing Model_5 with Andrwe_ng lecture Transcript



```
So for each example in Supervised Learning, we were told explicitly what is the so-called right answer, whether it's benign or malignant. In Unsupervised Learning, we're given data that looks different than data that looks like this that doesn't have any labels or that all has the same label or really no labels.
So we're given the data set and we're not told what to do with it and we're not told what each data point is. Instead we're just told, here is a data set. Can you find some structure in the data? Given this data set, an Unsupervised Learning algorithm might decide that the data lives in two different clusters. And so there's one cluster and there's a different cluster.


And yes, Supervised Learning algorithm may break these data into these two separate clusters.

So this is called a clustering algorithm. And this turns out to be used in many places.

One example where clustering is used is in Google News and if you have not seen this before, you can actually go to this URL news.google.com to take a look. What Google News does is everyday it goes and looks at tens of thousands or hundreds of thousands of new stories on the web and it groups them into cohesive news stories.

For example, let's look here.
The URLs here link to different news stories about the BP Oil Well story.
So, let's click on one of these URL's and we'll click on one of these URL's. What I'll get to is a web page like this. Here's a Wall Street Journal article about, you know, the BP Oil Well Spill stories of "BP Kills Macondo", which is a name of the spill and if you click on a different URL
from that group then you might get the different story. Here's the CNN story about a game, the BP Oil Spill,
and if you click on yet a third link, then you might get a different story. Here's the UK Guardian story about the BP Oil Spill.
So what Google News has done is look for tens of thousands of news stories and automatically cluster them together. So, the news stories that are all about the same topic get displayed together. It turns out that clustering algorithms and Unsupervised Learning algorithms are used in many other problems as well.
Here's one on understanding genomics.
Here's an example of DNA microarray data. The idea is put a group of different individuals and for each of them, you measure how much they do or do not have a certain gene. Technically you measure how much certain genes are expressed. So these colors, red, green, gray and so on, they show the degree to which different individuals do or do not have a specific gene.
Play video starting at :3:2 and follow transcript
And what you can do is then run a clustering algorithm to group individuals into different categories or into different types of people.
So this is Unsupervised Learning because we're not telling the algorithm in advance that these are type 1 people, those are type 2 persons, those are type 3 persons and so on and instead what were saying is yeah here's a bunch of data. I don't know what's in this data. I don't know who's and what type. I don't even know what the different types of people are, but can you automatically find structure in the data from the you automatically cluster the individuals into these types that I don't know in advance? Because we're not giving the algorithm the right answer for the examples in my data set, this is Unsupervised Learning.
Play video starting at :3:44 and follow transcript
Unsupervised Learning or clustering is used for a bunch of other applications.
Play video starting at :3:48 and follow transcript
It's used to organize large computer clusters.
Play video starting at :3:51 and follow transcript
I had some friends looking at large data centers, that is large computer clusters and trying to figure out which machines tend to work together and if you can put those machines together, you can make your data center work more efficiently.
This second application is on social network analysis.
So given knowledge about which friends you email the most or given your Facebook friends or your Google+ circles, can we automatically identify which are cohesive groups of friends, also which are groups of people that all know each other?
Market segmentation.
Many companies have huge databases of customer information. So, can you look at this customer data set and automatically discover market segments and automatically
group your customers into different market segments so that you can automatically and more efficiently sell or market your different market segments together?
Again, this is Unsupervised Learning because we have all this customer data, but we don't know in advance what are the market segments and for the customers in our data set, you know, we don't know in advance who is in market segment one, who is in market segment two, and so on. But we have to let the algorithm discover all this just from the data.
Finally, it turns out that Unsupervised Learning is also used for surprisingly astronomical data analysis and these clustering algorithms gives surprisingly interesting useful theories of how galaxies are formed. All of these are examples of clustering, which is just one type of Unsupervised Learning. Let me tell you about another one. I'm gonna tell you about the cocktail party problem.
So, you've been to cocktail parties before, right? Well, you can imagine there's a party, room full of people, all sitting around, all talking at the same time and there are all these overlapping voices because everyone is talking at the same time, and it is almost hard to hear the person in front of you. So maybe at a cocktail party with two people
```

Output



AI PROJECT

An NLP model to classify abstract sentences into the role they play (e.g. objective, methods, results, etc..) to enable researchers to skim through the literature and dive deeper when necessary.

Entre Abstract Here !!

that as a Supervised Learning problem just like we did for the breast tumor data.
So, that's it for Unsupervised Learning and in the next video, we'll delve more into specific learning algorithms and start to talk about just how these algorithms work and how we can, how you can go about implementing them.

□ Show Example Abstract

Extract !

Objective :

So, the news stories that are all about the same topic get displayed together. Octave, is free open source software, and using a tool like Octave or Matlab, many learning algorithms become just a few lines of code to implement.So when the video pauses, please answer the question on the slide. So, that's it for Unsupervised Learning and in the next video, we'll delve more into specific learning algorithms and start to talk about just how these algorithms work and how we can, how you can go about implementing them.

Background :

One example where clustering is used is in Google News and if you have not seen this before, you can actually go to this URL news.google.com to take a look.

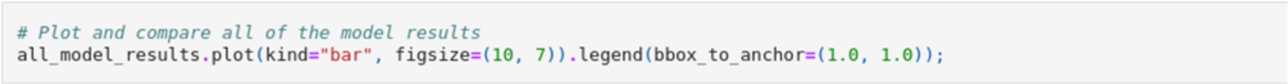
For example, let's look here. The URLs here link to different news stories about the BP Oil Well story.Here's a Wall Street Journal article about, you know, the BP Oil Well Spill stories of "BP Kills Macondo", which is a name of the spill and if you click on a different URL from that group then you might get the different story.I don't know who's and what type. Market segmentation.But we have to let the algorithm discover all this just from the data.All of these are examples of clustering, which is just one type of Unsupervised Learning. When you give these two microphone recordings to the same algorithm, what it does, is again say, you know, it sounds like there are two audio sources, and moreover, the album says, here is the first of the audio sources I found. It take researchers a long time to come up with this line of code.Or if you have Matlab you can use that too. And in fact what many people will do to in the large Silicon Valley companies is in fact, use an algorithm like Octave to first prototype the learning algorithm, and only after you've gotten it to work, then you migrate it to C++ or Java or whatever.

Methods :

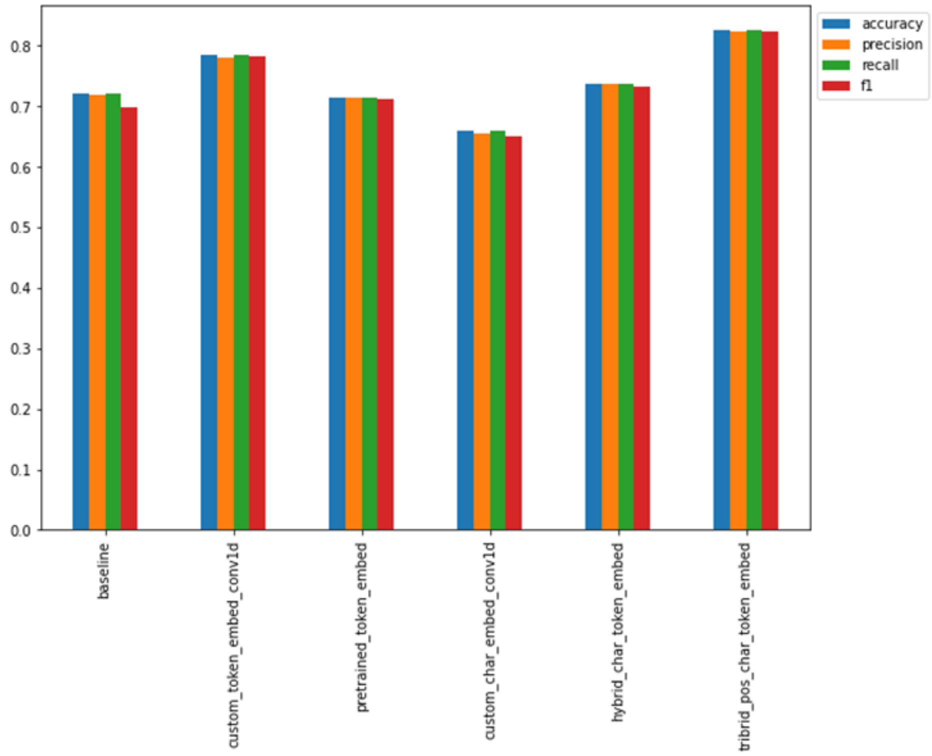
So for each example in Supervised Learning, we were told explicitly what is the so-called right answer, whether it's benign or malignant.Instead we're just told, here is a data set.

So this is called a clustering algorithm.And this turns out to be used in many places.What Google News does is everyday it goes and looks at tens of thousands or hundreds of thousands of new stories on the web and it groups them into cohesive news stories. So, let's click on one of these URL's and we'll click on one of these URL's.What I'll get to is a web page like this.Here's the CNN story about a game, the BP Oil Spill, and if you click on yet a third link, then you might get a different story.Here's the UK Guardian story about the BP Oil Spill. So what Google News has done is look for tens of thousands of news stories and automatically cluster them together. Here's one on understanding genomics. Here's an example of DNA

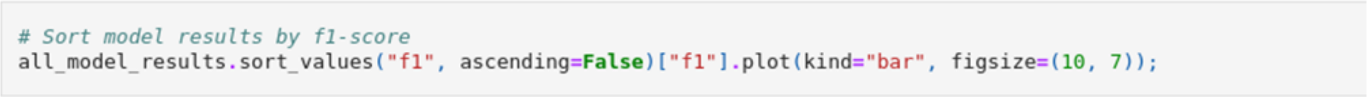
Plot with Accuracy, Precision, Recall, and F1 Score



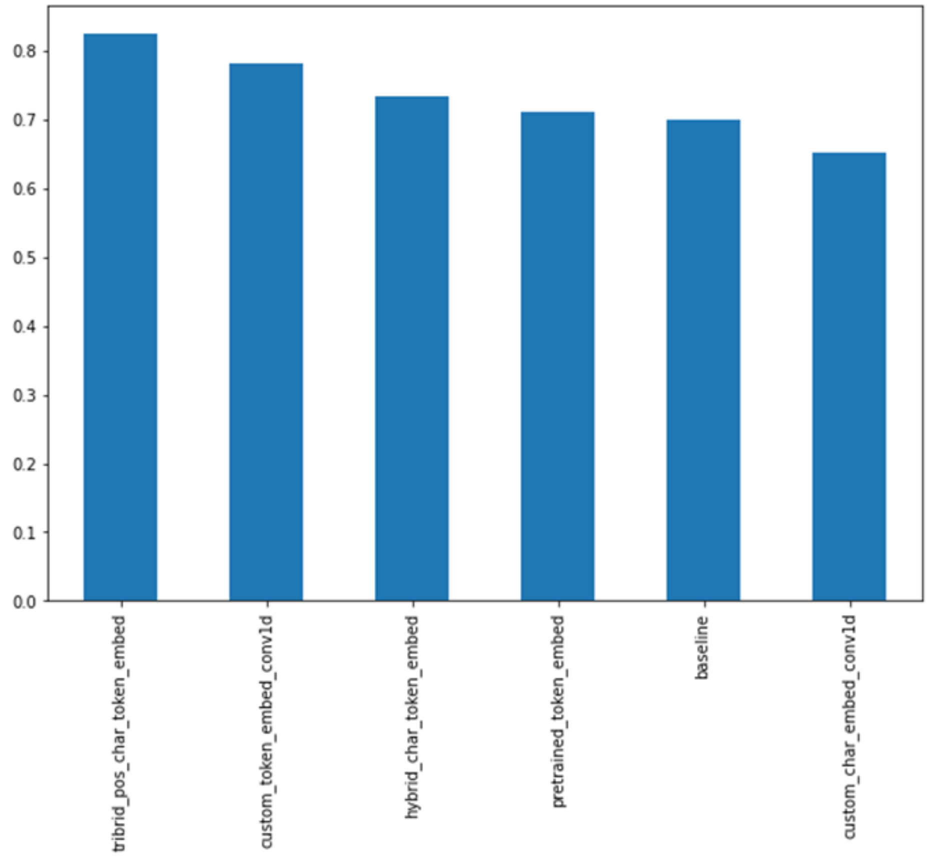
```
# Plot and compare all of the model results
all_model_results.plot(kind="bar", figsize=(10, 7)).legend(bbox_to_anchor=(1.0, 1.0));
```



Plot with an only F1 score

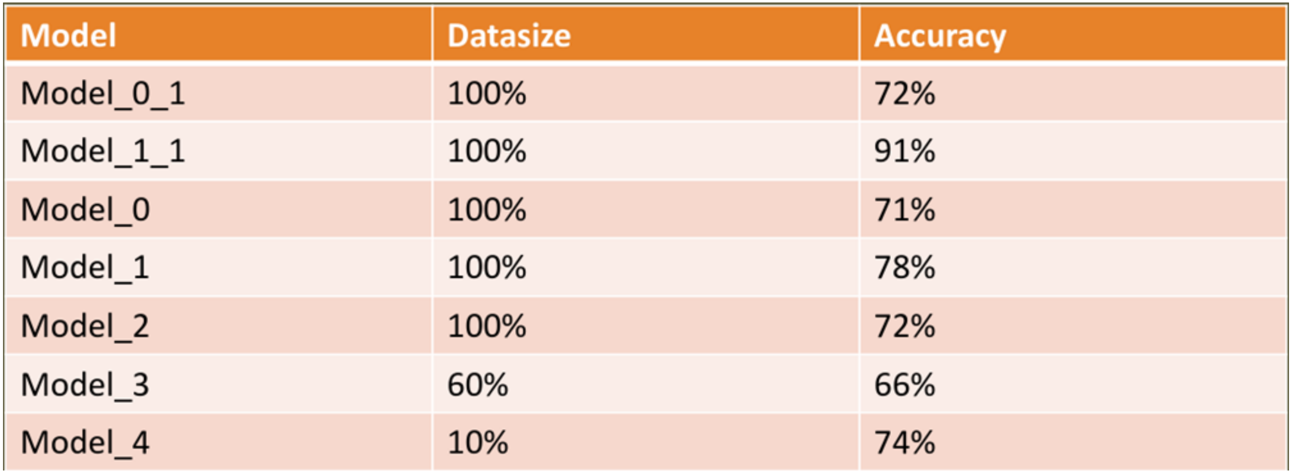


```
# Sort model results by f1-score
all_model_results.sort_values("f1", ascending=False)["f1"].plot(kind="bar", figsize=(10, 7));
```



Conclusion

Measuring Metrics of all the Models



Model	Datasize	Accuracy
Model_0_1	100%	72%
Model_1_1	100%	91%
Model_0	100%	71%
Model_1	100%	78%
Model_2	100%	72%
Model_3	60%	66%
Model_4	10%	74%

So, Model_1_1 has the best performance. While Model_3 has the worst performance.

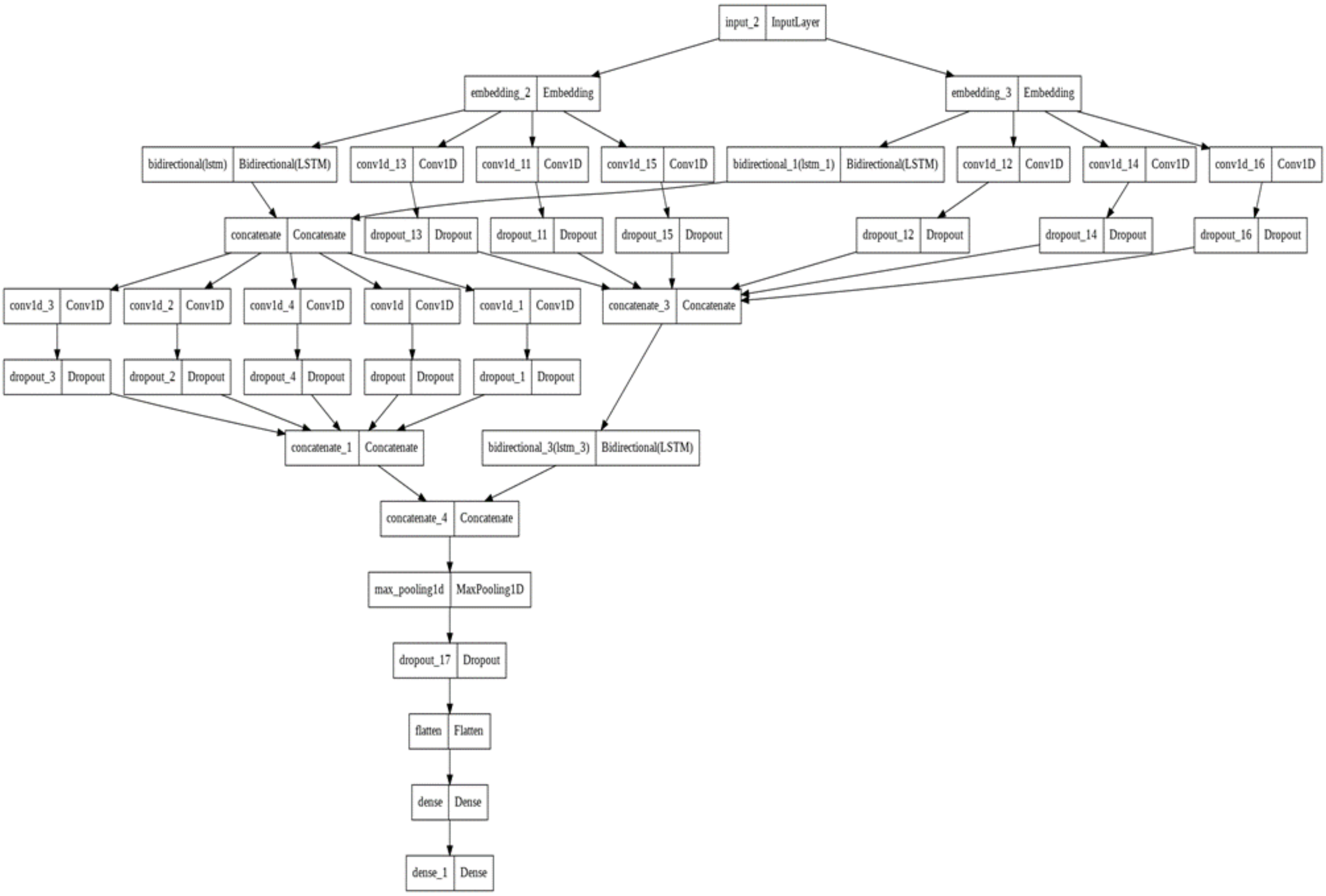
Future Scope of Works

a) The first most important is that all the models are trained with only 10% of the dataset because of GPU issues the F1 score of the models can be increased by increasing the epochs and with the whole datasets.

b) We have used pre-trained embedding from TensorFlow which will have an influence on the accuracy but there can be a case where some words like “Aphthous stomatitis” (it quite belonging to the medical field but a possibility with the regular words).

c) So, for the above problem, we have a Kind of idea of creating 2 separated Inputs layers for One layer for trained Words and another layer for untrained words.

Expected Model



Reference

1. Narendra Andhale and L.A. Bewoor (2016) An overview of Text Summarization techniques (1st edition)
2. Gabor Melis, Tomas Kocisky, and Phil Blunsom (2020) Mogrifier LSTM (2nd edition)
3. Elozino Egonmwan and Yllias Chali (2019) Transformer and seq2seq model for Paraphrase Generation (1st edition)
4. Ilya Sutskever, Oriol Vinyals, and Quoc V. Le (2014) Sequence to Sequence Learning with Neural Networks (3rd edition)
5. Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014) Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation (3rd edition)
6. Adhika Pramita Widyassari, Supriadi Rustad, Guruh Fajar Shidik, Edi Noersasongko, Abdul Syukur, Affandy Affandy, and De Rosal Ignatius Moses Setiadi (2020) Review of automatic text summarization techniques & methods (1st edition)
7. Deepali K. Gaikwad and C. Namrata Mahender (2016) A Review Paper on Text Summarization (1st edition)
8. Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut (2017) Text Summarization Techniques: A Brief Survey (3rd edition)
9. Ishitva Awasthi, Kuntal Gupta, Prabjot Singh Bhogal, Sahejpreet Singh Anand, and Piyush Kumar Soni (2021) Natural Language Processing (NLP) based Text Summarization - A Survey (1st edition)
10. Oguzhan Tas and Farzad Kiyani (2017) A survey automatic text summarization (1st edition)
11. Vipul Dalal and Latest Malik (2014) A Survey of Extractive and Abstractive Text Summarization Techniques (1st edition)
12. Ambedkar Kanapala, Sukomal Pal, and Rajendra Pamula (2019) Text summarization from legal documents: a survey (1st edition)
13. Prachi Shah and Nikita P. Desai (2016) A survey of automatic text summarization techniques for Indian and foreign languages (1st edition)
14. Franck Dernoncourt, Ji Young Lee, and Peter Szolovits (2016) Neural Networks for Joint Sentence Classification in Medical Paper Abstracts (1st edition)
15. S.A.Babara and Pallavi D.Patil (2015) Improving Performance of Text Summarization (2nd edition)
16. Pradeepika Verma and Anshul Verma (2020) A review on Text summarization Techniques (1st edition)
17. Neelima Bhatia and Arunima Jaiswal (2016) Automatic text summarization and its methods - a review (1st edition)