Automating Text Encapsulation Using Deep Learning

Anket Sah
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_projects

Part of the Artificial Intelligence and Robotics Commons

Recommended Citation
DOI: https://doi.org/10.31979/etd.rbxg-ehd6
https://scholarworks.sjsu.edu/etd_projects/995

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.
AUTOMATING TEXT ENCAPSULATION USING DEEP LEARNING

A PROJECT

Presented to
The Faculty of the Department of Computer Science
San Jose State University

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science

by
Anket Sah
May, 2021
The Designated Project Committee Approves the Project Titled

AUTOMATING TEXT ENCAPSULATION USING DEEP LEARNING

by
Anket Sah

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

San Jose State University

May 2021

Dr. Robert Chun Department of Computer Science
Dr. Navrati Saxena Department of Computer Science
Mr. Mayur Barge Software Engineer, Cisco
ABSTRACT

Data is an important aspect in any form be it communication, reviews, news articles, social media data, machine or real-time data. With the emergence of Covid-19, a pandemic seen like no other in recent times, information is being poured in from all directions on the internet. At times it is overwhelming to determine which data to read and follow. Another crucial aspect is separating factual data from distorted data that is being circulated widely. The title or short description of this data can play a key role. Many times, these descriptions can deceive a user with unwanted information. The user is then more likely to spread this information with his colleagues/family and if they too are unaware, this false piece of information can spread like a forest wildfire. Deep machine learning models can play a vital role in automatically encapsulating the description and providing an accurate overview. This automated overview can then be used by the end user to determine if that piece of information can be consumed or not. This research presents an efficient Deep learning model for automating text encapsulation and its comparison with existing systems in terms of data, features and their point of failures. It aims at condensing text percepts more accurately.

Keywords: Deep learning, Text Encapsulation, system, dataset, features, automated, model, machine learning
ACKNOWLEDGEMENTS

I would like to thank Dr. Robert Chun for his constant support and guidance throughout the research and implementation of this project. I would also like to thank my committee members Dr. Navrati Saxena and Mr. Mayur Barge for their inputs and valuable feedback on the project.

Lastly, I would like to thank my family and friends for their endless support and motivation along the way.
TABLE OF CONTENTS

I. Introduction .................................................................................................................................. 7
   1.1 Research Objective ................................................................................................................ 8
   1.2 Motivation .............................................................................................................................. 8

II. RELATED WORK .................................................................................................................... 10
   2.1 Scoring Sentences based on the Word-Frequency ............................................................... 12
   2.2 Universal Sentence Encoder for Text Ranking .................................................................... 12
   2.3 Unsupervised Learning using K-means Clustering ............................................................. 13

III. DATASET ................................................................................................................................. 14
   3.1 Amazon Reviews Dataset .................................................................................................... 14
   3.2 CNN News Dataset .............................................................................................................. 15

IV. DATA PREPARATION ............................................................................................................ 16
   4.1 Data Quality Issues .............................................................................................................. 16
      4.1.1 Manual Data Entry Errors ............................................................................................. 16
      4.1.2 Lack of complete information ....................................................................................... 16
   4.2 Contraction Mapping ........................................................................................................... 16
   4.3 Outlier Identification ............................................................................................................ 16
   4.4 Remove Duplicates and Filter Stopwords ............................................................................ 17

V. ALGORITHMS ......................................................................................................................... 18
   5.1 Naïve Bayes ......................................................................................................................... 18
   5.2 Extractive Summarization using BERT ............................................................................... 18
   5.3 TF-IDF .................................................................................................................................... 19
   5.4 Latent Semantic Analysis .................................................................................................... 19

VI. EVALUATION .......................................................................................................................... 21
   6.1 Evaluation Metrics ................................................................................................................ 21
VII. IMPLEMENTATION ................................................................................................................ 23
  7.1 Choosing the Corpus of Data ............................................................................................... 23
  7.2 Preprocessing / Data Cleaning ............................................................................................. 24
  7.3 Data Splitting ....................................................................................................................... 25
  7.4 Deep learning model ............................................................................................................ 25
    7.4.1 The Encoder .................................................................................................................. 26
    7.4.2 The Intermediate Vector ............................................................................................... 26
    7.4.3 The Decoder .................................................................................................................. 26
  7.5 Encapsulator ......................................................................................................................... 27
    7.5.1 Global Content Aware Consciousness .......................................................................... 28
    7.5.2 Local Content Aware Consciousness ............................................................................ 28
VIII. RESULTS AND OBSERVATIONS ........................................................................................ 33
  8.1 Approach 1 – Scoring Sentences based on the Word-Frequency ........................................ 34
  8.2 Approach 2 – Universal Sentence Encoder for Text Ranking ............................................. 36
  8.3 Approach 3 – Dual bi-directional LSTM ............................................................................. 37
  8.4 Model Training and Validation Loss ................................................................................... 42
  8.5 Run-Time Analysis .............................................................................................................. 44
  8.6 Comparison of the Results ................................................................................................... 45
IX. CONCLUSION AND FUTURE WORK ................................................................................... 48
References .......................................................................................................................................... 49
LIST OF TABLES

Table 1: Cosine Similarity Matrix 37
Table 2: Training and Validation Loss per Epoch for Scoring Sentences (Approach 1) 42
Table 3: Training and Validation Loss per Epoch for Text Ranking (Approach 2) 43
Table 4: Training and Validation Loss per Epoch for Dual LSTM (Approach 3) 43
Table 5: Run-time analysis 45
Table 6: Comparison between Three Approaches 46
LIST OF FIGURES

Figure 1: Unsupervised Learning using K-means 13

Figure 2: Amazon reviews Dataset (Headers: ProductId, UserId, Score, Time, Summary, Text) 15

Figure 3: CNN news dataset (Headers: Published Date, Author, Text, Source) 23

Figure 4: Deep Learning Model Architecture 25

Figure 5: System Depicting Encoder, Internal State and Decoder 30

Figure 6: Data flow in Encoder 30

Figure 7: Data Flow in Decoder 31

Figure 8: Deep Learning Model Network 32

Figure 9: Distribution of text length for Amazon Reviews Dataset 39

Figure 10: Distribution of text length for CNN Reviews Dataset 39
I. INTRODUCTION

Data is being generated and consumed in order of several terabytes each day. There is a dramatic increase in the influx of data sources. For a consumer airline, there is approximately half a terabyte of flight data that is generated during a single flight duration. As per research, by 2025 the number of connected devices is forecasted to reach 75 billion [12]. All these devices generate and consume data. A single user consumes approximately 2.9 GB of data per day which includes social media, news, scientific articles, banking and navigation data. As per Bloomberg, 69% of the data that is being consumed by an average user mainly consists of news articles, and for tech savvy audiences, this can go up to 87% [10].

As per New York Times, a user selects a news article to read based on the title or heading rather than the content [7]. The content itself can be misleading. For instance, an article distributed in the Times Daily under the title “The benefits of ginger for adults” was deluding. This article actually referenced that, “The investigation shows that ginger could probably affect serotonin and aid in concentration improvement but ginger’s chemical composition can severely affect dopamine exclusively due to the presence of phenolic compounds”. The article examines the slight possibility of the constructive outcomes of ginger on adults; however, the heading has an alternate meaning. A robust deep learning model to encapsulate this content, as well as automate the process that captures the exact holistic meaning of this content, becomes a necessity.
1.1 Research Objective

The goal of this research is to automate the text encapsulation process by making use of a dual LSTM (Long Short-Term Memory) encoder-decoder deep learning model. The current text summarizers are naïve and lack accuracy. A significant performance improvement can be attained by incorporating new modifications. These modifications include:

- improving the data pre-processing stage to attain accurate data free from outliers
- automating the text encapsulation for the entire cleaned dataset
- incorporating dual long short term memory network in the process pipeline
- performance comparison of the system developed with existing models

The intent in using a dual LSTM deep learning system is to encapsulate the text without changing the universal interpretation of the entire content. Also, the aim is to automate this task of encapsulation and reduce manual effort.

1.2 Motivation

If we take a look at the very fundamental level of computer science, we can find encapsulation in object-oriented programming and in object-oriented system design. Encapsulation simply means restricting the direct access to some of the object's components without changing the meaning or object’s functionality. In the real world too, depicting the exact meaning that the content carries are as important as encapsulation in object-oriented system design. Identifying the content and
accurately condensing it to form an appropriate heading is a crucial aspect of text encapsulation. Preserving the exact meaning of the entire content is of essence.

The present machine learning models require some amount of manual effort and many of these models simply compress the text by removing commonly used words. With the spread of covid-19, the blame game of governments has started. This has provided a platform for netizens to feed on fake news and irrelevant articles which might not be scientifically true. Among the overall audiences, 29% of these consumers include elders and some youth that are not scientifically literate and are more prone to spreading false information. With the unregulated and free use of social media, people even vent out their frustration on other users due to misleading content that is wrongly summarized and lure the users into reading that content.
II. RELATED WORK

Automating the text encapsulation using deep learning is a relatively new concept. There isn’t much research performed on this exact NLP problem, but there are related text summarization techniques explored over the years. In the year 2001, Nomoto et al. [1] made use of the C4.5 decision tree algorithm, generalized by the Concept of Minimal Definition Span (MDS) and equated it with unsupervised approaches. The approach had an error rate of 49%. This results in a distorted sentence sequence which is vaguely different in meaning than the actual content. The researcher Shuhua Liu, in a 2005 IEEE conference [2] suggested a two-phase topic guided text condensation technique. In the first phase, passages are extracted from a public file, and in the second phase, text comprehension and merging of data is performed. This merging is assisted by syntactic and semantic tools to form a meaningful description. This paper provides better accuracy for a single keyword extraction but performs at 57% accuracy for key phrase extraction and summarization.

Zhang et al [3] describe a three-step approach for text summarization in the 2009 IEEE CSIT conference. They first grouped the sequences in the text; then measured the total statistical correlation on every group depending on the multi-feature set; and finally picked the subject phrase by following their established guidelines. The researchers performed their experiment on the DUC2003 document dataset. Nomoto et al. [1] attained an F1 measure of 0.432 and Zhang et al [3] achieved a score of 0.475. This means that this approach [3] can retain the meaning of summarized text with 21% higher accuracy than the approach described by Nomoto [1]. Thakkar
et al. [4] take a graph-based approach for summarizing text. The researchers suggest the use of a shortest path algorithm as it creates a seamless text snippet as opposed to keyword scoring algorithms from the previous approaches.

Another graph-based approach for text summarization is discussed by Reategui et al. [5]. This approach uses a text mining tool named Sobek that is developed using an N-simple version of an interval network. This network consists of nodes that include key words given in the text, and the edges represent data regarding adjacency. Sobek tool counts the total occurrences of each word, assigns a weight to those words and generates a summary that is somewhat in line with the content. On the other hand, in 2014, Ferreira et al. [6] suggested combining the sentence scoring methods for summarizing text. The two approaches for consolidating the sentence scoring techniques discussed in the paper include: (I) By Ranking: Every assistance chooses the primary text sequence and the client consolidates it in some way or another; (II) By Accentuation: The administration scores every text sequence and returns one sequence with refreshed scores. The accuracy for this approach wherein the expected output matches the actual output is 78%.

One of the other approaches to solving the problem of text summarization is an extractive summarization. Moratanch et al. [7] perform a survey on all the extractive summarization methods and put forward their results. The basic principle behind the working of extractive summarization is that it depends on the retrieval of multiple sections from a sequence of text, such as sentence sequences and its dependent phrases, and combine everything with each other to create a short description.
For example, given a text sequence: Covid-19 has spawned multiple global health crisis some have dubbed coronasomnia -- an inability to fall asleep or get good quality sleep during the pandemic. Along with this, there are multiple levels of stress associated with the pandemic – financial, health care related and social isolation. All these damage mental health, threaten health and quality of life for upto 45% of world’s population. (source CNN news dataset)

Extractive summary: ‘Coronasomnia’ -- an inability to fall asleep

The terms have been derived and merged to generate a description as seen above, but the interpretation can be syntactically and grammatically odd.

2.1 Scoring Sentences based on the Word-Frequency

This technique assigns weights to every single word that occurs in the text. For example, if the word ‘research’ appears 5 times in the text body, a weight of 5 is assigned to the word. Similarly, a holistic score based on the word appearance is assigned to every sentence that occurs in the text body. The sentences that have a higher score are then picked up and displayed as summarized text.

2.2 Universal Sentence Encoder for Text Ranking

Text Ranking is similar to PageRank used by Google. PageRank creates a matrix of pages that will be most likely visited next by the user. Similarly, TextRank uses cosine function to determine the similarity of 2 sentences to each other. This cosine linear similarity matrix is then
used to build a tree. The PageRank rating equation is then added to the tree in order to determine rankings for each statement.

### 2.3 Unsupervised Learning using K-means Clustering

Every text has some central theme around which the content revolves. This theme word appears multiple times in the text and helps understand what the text is about. This word is taken as a center, weight is assigned to it and the nearest K-words are chosen from the clusters. These K-words are then displayed as summarized text. This method has high error rate as the training data is unlabeled.

![Figure 1: Unsupervised Learning using K-means](image_url)
III. DATASET

Two datasets that include the CNN News Dataset and the Amazon reviews dataset, provide an accurate set of input data required to train the model. Here are their descriptions:

3.1 Amazon Reviews Dataset
The researchers from the Department of Computer Science at the University of California, Berkeley have created a text summarization dataset in 2020. During the covid-19 pandemic, purchase of essentials such as toilet papers, tissues, eateries skyrocketed due to public induced fear of lockdown. This is a reason why it can provide a good source of most recent data from public, news houses and scientists about the products used by them daily. This dataset contains Amazon ratings and reviews of all the stated essentials purchased during the covid pandemic. Additionally, the dataset includes data that spans more than a decade, with all 1 million reviews up to October 2012 included. Product and usage stats, scores, and a simple text summary are also used in reviews. It also contains ratings from all of Amazon's other categories.

The following are some of the characteristics of this dataset that make it suitable for this project:

- The dataset is sufficiently broad for the algorithm to be efficiently trained. A model's performance improves when it is exposed to heterogenous data.
- The model’s encapsulation will improve as the dataset contains unrelated user data and reviews.
- The reviews are simple text, meaning that most scripting frameworks can tokenize them.
3.2 CNN News Dataset

This is a custom dataset that has been created by scrapping news data from CNN using a python script. The url used to scrape data (https://www.cnn.com/specials/world/coronavirus-outbreak-intl-hnk) contains most recent world news related to Covid-19 pandemic. The dataset comprises of over 10,000 records, with an average of 400 words per record. The characteristics of this dataset that make it suitable for this project include:

- Large number of words per record that helps test the performance of model as the dataset size increases.

- A combination of news data from different categories that include covid-19 pandemic, national news, sports and lifestyle news.
IV. DATA PREPARATION

4.1 Data Quality Issues

4.1.1 Manual Data Entry Errors

People are prone to making mistakes, and a simple set of data containing manually entered data is likely to produce errors. Typos, data entered in the incorrect sector, missing entries, and other mistakes are almost unavoidable.

4.1.2 Lack of complete information

There can be some records in the dataset with missing fields. For example, in a dataset containing the username and reviews posted by that user, the reviews or username might be missing which makes the dataset lack vital information. These fields can be filled with null values or the entire record can be removed.

4.2 Contraction Mapping

The news articles and reviews are raw thoughts expressed by the writer. This is why it can have slangs and words like isn’t, needn’t, shan’t and so on. The root words are required to have a well-trained model. As a result, these slangs and compressed words must be mapped to actual words. For example, isn’t → is not, needn’t → need not, shan’t → shall not and so on.

4.3 Outlier Identification

An outlier is a particular occurrence that tends to differ significantly from the rest of the data. The following are some of the reasons why finding possible outliers is crucial. An outlier may be a sign of skewed results. For instance, data may have been wrongly interpreted, or a text could contain numbers and special characters.
4.4 Remove Duplicates and Filter Stopwords

Drop_duplicates() and dropna() functions help eliminate duplicates. Natural language toolkit contains stop words such as commonly occurring pronouns which get filtered out. Converting all text to lowercase, splitting individual strings to aid tokenization are some methods that are used to prepare data before training the model.
V. ALGORITHMS

5.1 Naïve Bayes

Naive bayes algorithm is based on the rule that all features of each category are independent.

The stages involved in Naïve Bayes summarization include:

1. Extract features
2. Count the number of features
3. Calculate probability
4. Generate Summary

Each individual sentence is a feature in each category. It calculates the number of features in each category and records this amount. A weight is then assigned to each sentence. Each Key is an independent sentence, and value is the number of times that sentence occurs in the text.

For example, {“sentence_1”, 2} means that “sentence_1” appears 2 times in the whole text and is assigned more weight.

Next it calculates the probability of feature occurrence in each category. Based on this relative probability, a summarization for a piece of text is generated.

5.2 Extractive Summarization using BERT

The job of extractive summarization is a daunting one which has gradually emerged as feasible. One explanation for this development, as with many items in NLP, is the advanced mappings
provided by converter templates like BERT. Using BERT sentence encoding and two supervised models, this method builds an extractive summarizer. Only encodings and their variations are considered in the first model. This is consistent with an intuition that a strong parser could parse context and can pick sentences solely based on the article's inner core. Another model uses data patterns and builds on the popular Lead3 phenomenon, which is unique to newspaper corpora. The Lead3 phenomenon is based on the concept that the first three sentences of an essay usually summarize it well. Most authors, in reality, implement this technique directly.

5.3 TF-IDF

TF-IDF stands for Term Frequency – Inverse Document Frequency. This algorithm summarizes articles and text based on the weight assigned to each word in the document. It weighs down commonly occurring words like he, the, they, them and all the pronouns and weighs up less frequently occurring words. Based on the weights of each of the words, top weighted words are chosen and a summary is generated by using those words. This summary is more focussed on the keyword extraction rather than capturing the holistic meaning of the text. This is why the summary generated can be very vague in meaning as compared to the original text.

5.4 Latent Semantic Analysis

Latent semantic examination (LSA) is an algorithm for extracting a portrayal of text articles dependent on the noticed words. The initial step is to construct a term-sentence grid, where each line is a word from the data (n words) and every segment is a sentence. Every passage of the grid
is the heaviness of the word i in sentence j processed by the TFIDF method. Now, Solitary Val Decomposition (SVD) is utilized on the grid that changes the underlying grid into three grids: a term-theme grid having loads of words, a corner-to-corner grid where each column relates to the heaviness of a subject, and a point sentence grid. By increasing the corner-to-corner grid with loads and the subject sentence network, the outcome will portray how much a sentence represents a theme, in other words, the heaviness of the topic i in sentence j.
VI. EVALUATION

6.1 Evaluation Metrics

One of the difficulties in evaluating encapsulated text is that it needs the presence of a collection of reference, or short descriptions. These aren't typically accessible for many of these articles, which is why analysis is dominated by newspaper texts and academic articles. Research journals have manuscripts, whereas media outlets typically use roundups or banners for their pages. To evaluate the performance of the system developed, a combination of bleu measures to measure the preciseness and rouge measures to measure recall can be used.

Bleu measures precision: how many terms (n-grams) from the computer-generated encapsulations are present in the human description summaries.

Rouge measures recall: how many terms (n-grams) from the human description summaries are present in the computer-generated encapsulations.

These outputs closely accompany each other. If there are multiple n-grams from the computer output present in the human summaries, the Bleu is high. If there are multiple n-grams from the human summaries present in the computer output, the Rouge is high.

To address this, there is a concept named brevity penalty that can be added to Bleu implementations. It imposes a penalty on computer outcomes that are less than a reference's average length. This is in comparison to the n-gram parameter method, which incentivizes longer than reference outcomes by increasing the denominator as the computer results gets longer.
Finally, to make the metrics work together, an F1 measure is calculated given by the formula:

\[ F1 = 2 \times \frac{\text{Bleu} \times \text{Rouge}}{\text{Bleu} + \text{Rouge}} \]

An alternative evaluation method can use the number of sentences against the total number of sentences to validate results, for example,

\[ \text{Precision} = \frac{\text{Number of important sentences}}{\text{Total number of sentences summarized}}. \]

\[ \text{Recall} = \frac{\text{Total number of important sentences Retrieved}}{\text{Total number of important sentences present}}. \]

\[ \text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \]

\[ \text{Compressed Rate} = \frac{\text{Total number of words in the summary}}{\text{Total number of words in original document}}. \]
VII. IMPLEMENTATION

7.1 Choosing the Corpus of Data

Choosing the right data set is a crucial step in creating a consistent deep learning system. The very first step is to consolidate data from different sources. These include Amazon reviews dataset and a CNN news dataset. The CNN news dataset is created by scrapping news with the help of a python script. This dataset contains world news pertaining to the topic of Covid-19. These 2 datasets are ideal for testing the model as they contain most recent data that is entered by users, journalists, and writers. It covers covid news and reviews of products used by people during lockdowns. As this data covers a wide range of topics, it is heterogeneous and can test the developed model well.

![Figure 3: CNN news dataset (Row Headers: Published Date, Author, Text, Source)](image-url)
7.2 Preprocessing / Data Cleaning

This is a very important step in creating a fault-tolerant deep learning system which is mostly ignored by professionals. The price paid is a deep learning model that considers outliers such as blank spaces and special characters as important aspects and keeps including them in the output (which should not be included in the ideal output). This is why it is of utmost importance to eliminate the blank spaces and special characters. To clean the content, the following steps are involved that include:

- Eliminating additional void areas
- Expanding Contractions
- Eliminating special characters and uncommon strings
- Converting all characters to lowercase

This data pre-processing can be performed using some pre-existing tools such as openrefine or weka. However, these tools fail to perform well in case of the dataset used in this system. Openrefine and weka tools can handle small datasets and fail in case of large datasets. These tools throw an outOfMemoryError() when the dataset size increases beyond kilobytes. The developed system makes use of NLP toolkit that consists of stop words and a custom coded function. The NLP toolkit removes pronouns and commonly occurring repeated words as it encounters them. The custom coded function converts all uppercase letters to lowercase, eliminates blank spaces and deletes special characters on parsing the input.
7.3 Data Splitting

Preprocessing provides a dataset that is free from outliers. This data can now be divided into 2 sets:

- Training data set
- Test data set

The dataset division is done in such a manner that 75% of input data is chosen as training dataset and the remaining 25% dataset is chosen as the test dataset. If we use the entire dataset that is 100% as a training dataset, then there can be a problem of overfitting. Overfitting happens when a function is prepared too well on a restricted arrangement of information. At that point, when a model gets trained with so much information, it begins learning from the noise and inaccurate information. For example, consider a model that is trained to detect animals. But if this model is trained only on the images of cats and dogs and we pass an image of black bear during evaluation, the model will classify that bear as a dog.

7.4 Deep learning model

The deep learning model developed, consists of an encoder-decoder architecture at its heart.

![Deep Learning Model Architecture](image)

**Fig. 4: Deep Learning Model Architecture**

This model comprises three fundamental components that include: the encoder, an intermediate vector and the decoder.
7.4.1 The Encoder
The encoder essentially includes a sequence of stacked neurons from LSTM. The training set is taken by the encoder and this data is summarized into internal state sequences. The decoder then utilizes the sources of the encoder and the series of inner states. The input data is a list of all the keywords from the content that must be condensed in our text encapsulation system. Each keyword is portrayed as \( k_i \) where the order of this keyword is \( i \).

7.4.2 The Intermediate Vector
This is the actual concealed state created from the model's encoder. It is processed utilizing the equation (1) given below. In efforts to support the decoder to make valid inferences, this variable (vector) attempts to summarize the data for all input components. It functions as the initial concealed portion of the model that makes up the decoder.

7.4.3 The Decoder
The decoder consists of a multi-recurrent system array where each array calculates a value \( y_t \) at time phase \( t \). Each multi-recurrent unit embraces a concealed state and outputs a result along with its own concealed state from the previous module. The result stream is a list of all the keywords from the condensed content in the developed encapsulation system. Each keyword is denoted as \( y_i \), where the order of this keyword is \( i \).

For computing any initial concealed state \( h_i \), the formula in [13] is used:

\[
h_t = f(W^{(hh)}h_{t-1})
\]  

(1)
Here, it is evident that a prior concealed state is used to calculate the next one.

The output $y_t$ at a time instant $t$ can be calculated using the formula in [13]:

$$y_t = \text{softmax}(W^S h_t)$$  \hspace{1cm} (2)

The result is then computed by utilizing the concealed state at the present time phase along with the associated weight $W^s$. Softmax is utilized to generate a likelihood vector which will assist us in deciding the final result (for example, a single keyword answer in the question-answering problem).

### 7.5 Encapsulator

The DL model then generates an encapsulator that is capable of condensing the content without changing the exact meaning of the text. Its performance can be tested using the test dataset. The encapsulator generated has a content aware consciousness that plays a key role in keeping the meaning of the text intact.

Content Aware Consciousness (CAC): The key idea behind this mechanism is how much focus should be kept on every phrase in the input data so as to create a keyword at time phase $t$. For example:

**Question:** What are the factors that have impacted mental health during the pandemic?

**Answer:** Covid-19 has spawned multiple global health crisis some have dubbed coronasomnia -- an inability to fall asleep or get good quality sleep during the pandemic. Along with this, there are multiple levels of stress associated with the pandemic –
financial, health care related and social isolation. All these damage mental health, threaten health and quality of life for upto 45% of world’s population. (source CNN news)

In the inquiry above, the 12th word ‘pandemic’ is related to ‘Covid-19’ and the 4th word ‘factors’ is related to ‘Coronasomnia’, ‘financial’, ‘health’, ‘social isolation’.

In this way, rather than taking a gander at all the keywords in the input arrangement, the significance of explicit pieces of the text can be emphasized that generate the ideal result. This is the fundamental idea behind the Content Aware Consciousness.

Based on the manner in which the background variable is extracted, there are 2 distinct classes of consciousness:

7.5.1 Global Content Aware Consciousness

The emphasis is laid on all the positions (all sentences) of the source. In other terms, for extracting the supported context variable, all the concealed states of the encoder are taken into account. The system uses this global content aware consciousness.

7.5.2 Local Content Aware Consciousness

The emphasis is laid on only a few positions (2-5 crucial sentences) of the source. For extracting the supported context variable, only a few concealed states of the encoder are taken into account.

This is how the CAC works:

- The encoder generates the concealed state \( h_j \) for each time phase \( j \) in the source series
- Likewise, the decoder generates the concealed state \( s_i \) for each time phase \( i \) in the target series
• The arrangement score ($e_{ij}$) is calculated based on the source series which is aligned with the target series using a function for score estimation. The arrangement score is calculated from the source concealed state $h_j$ and target concealed state $s_i$ using the score function, which is given by:

$$e_{ij} = \text{score}(s_i, h_j)$$

where $e_{ij}$ indicates the arrangement score for target time phase $i$ and source time phase $j$.

• The arrangement scores are then normalized by using the softmax function to extract the CAC weights ($a_{ij}$):

$$a_{ij} = \frac{e_{ij}}{\sum_{k=1}^{T_x} e_{ik}}$$

• Now the CAC context variable ($C_i$) is calculated from the summation of CAC weights and concealed states of encoder $h_j$:

$$C_i = \sum_{j=1}^{T_x} a_{ij} h_j$$

• The concealed variable $V_i$ is generated by integrating the CAC context variable and the concealed state of the decoder at time phase $i$:

$$V_i = \text{integrate}(s_i, C_i)$$

• To create the final output $y_i$, the concealed variable $V_i$ is further loaded into the deep network:

$$y_i = \text{deepNet}(V_i)$$
Figure 5: System Depicting Encoder, Internal State and Decoder

With the aid of an example, let's comprehend the above-mentioned CAC process. Consider \([x_1, x_2, x_3, x_4]\) as the input content and \([y_1, y_2]\) as the output encapsulated text.

- For each time phase \(t\), the encoder scans the complete input series and produces a set of concealed states \(h_1, h_2, h_3, h_4\)

Figure 6: Data Flow in Encoder

- The decoder scans one time phase offset of the complete target series, and produces the concealed state for each time phase \(s_1, s_2, s_3\)
The arrangement scores $e_{1j}$ are determined using the score function from the concealed input state $h_i$ and target concealed state $s_1$:

$$e_{11} = \text{score}(s_1, h_1)$$
$$e_{12} = \text{score}(s_1, h_2)$$
$$e_{13} = \text{score}(s_1, h_3)$$
$$e_{14} = \text{score}(s_1, h_4)$$

The arrangement scores are then normalized by using the softmax function to extract the CAC weights $(a_{ij})$:

$$a_{11} = \frac{e_{11}}{e_{11} + e_{12} + e_{13} + e_{14}}$$
$$a_{12} = \frac{e_{12}}{e_{11} + e_{12} + e_{13} + e_{14}}$$
$$a_{13} = \frac{e_{13}}{e_{11} + e_{12} + e_{13} + e_{14}}$$
$$a_{14} = \frac{e_{14}}{e_{11} + e_{12} + e_{13} + e_{14}}$$

Now the CAC context variable $(C_i)$ is calculated from the multiplication of CAC weights $a_{ij}$ and concealed states of encoder $h_j$

$$C_2 = h_1a_{21} + h_2a_{22} + h_3a_{23} + h_4a_{24}$$
Concealed variable $V_i$ is generated by integrating the CAC $C_2$ variable and the concealed state $s_2$

$$V_2 = \text{integrate}([s_2; C_2])$$

The concealed variable $V_2$ is further loaded into the deep network to generate output $y_2$

$$y_2 = \text{deepNet}(V_2)$$

$y_3$, $y_4$ and so on are calculated in a similar fashion as shown below.

---

Model: "model"

<table>
<thead>
<tr>
<th>Layer (type)</th>
<th>Output Shape</th>
<th>Param #</th>
<th>Connected to</th>
</tr>
</thead>
<tbody>
<tr>
<td>input_1 (InputLayer)</td>
<td>[(None, 80)]</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>embedding (Embedding)</td>
<td>(None, 80, 500)</td>
<td>2558500</td>
<td>input_1[0][0]</td>
</tr>
<tr>
<td>lstm (LSTM)</td>
<td>[(None, 80, 500), (None, 2002000)]</td>
<td>537500</td>
<td>embedding[0][0]</td>
</tr>
<tr>
<td>input_2 (InputLayer)</td>
<td>[(None, None)]</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>lstm_1 (LSTM)</td>
<td>[(None, 80, 500), (None, 2002000)]</td>
<td></td>
<td>lstm_0[0][0]</td>
</tr>
<tr>
<td>embedding_1 (Embedding)</td>
<td>(None, None, 500)</td>
<td></td>
<td>input_2[0][0]</td>
</tr>
<tr>
<td>lstm_2 (LSTM)</td>
<td>[(None, 80, 500), (None, 2002000)]</td>
<td></td>
<td>lstm_1[0][0]</td>
</tr>
<tr>
<td>lstm_3 (LSTM)</td>
<td>[(None, None, 500), 2002000]</td>
<td></td>
<td>embedding_1[0][0]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>lstm_2[0][1]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>lstm_2[0][2]</td>
</tr>
</tbody>
</table>

Figure 8: Deep Learning Model network
VIII. RESULTS AND OBSERVATIONS

To evaluate the output of each of the approaches, a recent news snippet is chosen with over 1250 words, related to Covid-19 as shown below:

[CNN] The US will likely soon come up against a significant obstacle in the fight against the Covid-19 pandemic -- more doses of the vaccines than people who are willing to receive them, according to data that is worrying experts. "We actually think that (vaccine) supply will outstrip demand ... in probably the middle of May," Dr. Chris Murray, chair of the Institute for Health Metrics and Evaluation (IHME) at the University of Washington, told CNN Friday. The foundation cited the percentage of survey respondents that said they’ve already received the vaccine or would get it as soon as they could – 62%. Weighing that against the daily rate of vaccinations, and the CDC’s data showing more than 41% of US adults already have had at least one dose, it’s possible that the vast majority of US adults who presently want a first dose will have received them by early-to-mid-May, the KFF said. One modifying factor, the foundation said, would be how many of the people who say they’ll "wait and see" -- 13% -- decide to get a vaccine in the next few weeks. Murray cited different data -- daily Facebook surveys, watched by his institute, that he said showed a decline in percentage of adults wanting the vaccine since February: About 67%, down from around 75%. But both Murray and KFF have said their different indicators point to a significant slowdown in new vaccinations sometime in May -- and experts have said more people will need to be persuaded to get one, and more people will need to be made eligible, to be assured of herd immunity. Health officials -- including Dr. Anthony Fauci -- estimate that somewhere between 70% to 85% of the country needs to be immune to the virus -- either through inoculation or previous infection -- to suppress its spread. Currently, no Covid-19 vaccine is authorized in the US for people younger than 16. Vaccine makers have been studying their efficacy and safety in younger people, and Pfizer has asked the FDA to authorize its product for ages 12-15. Expanding vaccine eligibility will be key to stamping out the pandemic, Dr. Leana Wen told CNN. "It's going to be very difficult, if not impossible, for us to reach herd immunity unless our children are also vaccinated," Wen, a CNN medical analyst and a former Baltimore health commissioner, said Saturday. The average number of Covid-19 vaccinations administered in the US per day has been falling recently. The average was 2.86 million a day as of Friday morning, down from a record 3.38 million on April 13, according to CDC data. Already, the military is seeing a surplus of doses and a steady decline in the rate at which they are used. "We have heard anecdotally that younger people may feel that they're not as vulnerable to Covid and that perhaps the risk of getting vaccination is higher than getting the disease, which of course we know not to be true," Terry Adirim, acting assistant defense secretary for health affairs, told reporters this week. Model predicts declining deaths, but vaccine hesitancy could hurt Daily Covid-19 deaths in the US should continue to decline in the coming months, though that could change if vaccine hesitancy rises, Murray's IHME said Friday. The country averaged 704 Covid-19 deaths a day across the last week, according to Johns Hopkins University data -- down from the 3,000s in parts of January and February. An IHME model projects that average will dip to around 425 by June 1, and 105 by August 1, the institute said Friday. Recent vaccine expansion and mask usage have helped death rates decline even as more contagious coronavirus variants -- such as B.1.1.7 -- have spread, the IHME said. But in a worst-case scenario, daily deaths could stay in the 700s in May and June, and in the 600s through July, the IHME predicts. That scenario involves a fast decline in mask use and an increase in mobility, the institute said. Vaccine hesitancy also would be troublesome, the IHME said. "Given how central vaccination is to the US strategy to control the B.1.1.7 potential surge, the slow erosion of vaccine confidence unfolding over the last two or more months is cause for concern," the IHME said. Lifting of J&J pause is welcome news, CDC director says On Friday, the CDC and the Food and Drug Administration lifted their recommended pause on the use of the Johnson & Johnson vaccine. The agencies had recommended the pause April 13 after learning of cases a rare blood
clotting syndrome among women who had recently received the vaccine. After examining data about the cases, the CDC’s vaccine advisory committee voted to recommend lifting the pause, essentially deciding the benefits of the vaccine outweighed the risks. But the FDA will update the vaccine’s label, indicating women under the age of 50 should be aware of a risk of the rare blood clotting syndrome linked to that product. The CDC said it has collected reports of 15 such cases, all in women and 13 of them in women under 50. At most, resuming administration of the J&J vaccine would result in a few dozen rare blood clots while saving hundreds of lives, a CDC analysis showed. “When resuming vaccination among all persons at least 18 years, we expect 26 to 45 TTS cases depending on vaccine uptake,” CDC’s Dr. Sara Oliver said, referring to the rare blood clots known as thrombosis-thrombocytopenia syndrome. But 600 to 1,400 deaths from Covid-19 would be prevented, and as many as 3,500 ICU admissions would be prevented. The CDC’s director, Dr. Rochelle Walensky, said Friday the lifted pause is “welcome news for many, as many have wanted the Johnson & Johnson vaccine to fill an important need in vaccination efforts here and around the world.” “I think we have to do extraordinary outreach to clinicians – as we have been doing this past week; we already have plans to start that on Monday to public health officials -- and then we have to do extraordinary outreach to patients, to meet people where they’re at, to educate them” about Covid-19 vaccines, Walensky said at a joint conference held by the CDC and the FDA. “Overall, I actually think that this pause conveyed that we are taking every one of these needles in haystacks that we find seriously – that we’re really examining, scrutinizing the data that we’re seeing.” The Moderna and Pfizer/BioNTech vaccines have not been associated with blood clots, a CDC’s vaccine advisory committee was told Friday Study shows how to reduce infections in children While vaccines remain unavailable to children, new research suggests that testing in school and the vaccination of adults may lower infections in children. So far, no vaccines are authorized for people younger than 16, but a study published in the medical journal JAMA Network Open on Friday showed that quickly identifying and contact tracing children to identify “silent infections” of Covid-19, where the disease is either presymptomatic or asymptomatic, combined with vaccination of 40-60% adults could significantly reduce the amount of disease. In a different scenario, where silent infections remained undetected, researchers estimated that children would need an 81% vaccination rate, in addition to 40% of adults being vaccinated, in order to achieve a similar infection rate. As more students return to the classroom, the study provides a road map to continuing reducing the spread of the virus even before children are eligible for the shots.

Original Title: No vaccine for younger adults despite of excessively produced number of doses

8.1 Approach 1 – Scoring Sentences based on the Word-Frequency

This technique assigns weights to every single word that occurs in the text. For example, if the word ‘research’ appears 5 times in the text body, a weight of 5 is assigned to the word. Similarly, a holistic score based on the word appearance is assigned to every sentence that occurs in the text body. The sentences that have a higher weight are then picked up and used to summarize the text.
As shown in the snippet above, the words are tokenized, weights are assigned to all the words and the sentence score is computed. To generate the overview, we'll take the top N statements which have a score higher than all the other statements. These statements are then chosen to generate the encapsulations.
AUTOMATING TEXT ENCAPSULATION USING DEEP LEARNING

The generated encapsulated text:

Currently, no Covid-19 vaccine is authorized in the US for people younger than 16. Vaccine makers have been studying their efficacy and safety in younger people, and Pfizer has asked the FDA to authorize its product for ages 12-15. Expanding vaccine eligibility will be key to stamping out the pandemic, Dr. Leana Wen told CNN. The average number of Covid-19 vaccinations administered in the US per day has been falling recently. An IHME model projects that average will dip to around 425 by June 1, and 105 by August 1, the Institute said Friday. Recent vaccine expansion and mask usage have helped death rates decline even as more contagious coronavirus variants -- such as B.1.1.7 -- have spread, the IHME said. That scenario involves a fast decline in mask use and an increase in mobility, the Institute said. Vaccine hesitancy also would be troublesome, the IHME said. After examining data about the cases, the CDC’s vaccine advisory committee voted to recommend lifting the pause, essentially deciding the benefits of the vaccine outweighed the risks. The CDC said it has collected reports of 15 such cases, all in women and 13 of them in women under 50. But 600 to 1,400 deaths from Covid-19 would be prevented, and as many as 3,500 ICU admissions would be prevented.

8.2 Approach 2 – Universal Sentence Encoder for Text Ranking

Text Ranking is similar to PageRank used by Google. PageRank creates a matrix of pages that will be most likely be visited next by the user. Similarly, TextRank uses cosine function to determine the similarity of 2 sentences to each other. This cosine linear similarity matrix is then used to build a tree. The PageRank rating equation is then added to the tree in order to determine rankings for each statement.

```python
from sklearn.metrics.pairwise import cosine_similarity
import networkx as nx

# To create the cosine similarity matrix
mx = cosine_similarity(message_embeddings)

# To generate graph and scores from the textrank method
nx_gh = nx.from_numpy_array(mx)
score = nx.pagerank(nx_gh)

stm_rank = sorted(((score[i], s) for i, s in enumerate(stm)), reverse=True)
num_of_stm = 2

result = " ".join([i[1] for i in stm_rank[:num_of_stm]])
print(result)
timeElapsed = datetime.now() - startTime
print('Time elapsed (hh:mm:ss.ms) {}\'.format(timeElapsed))
```
The table below shows the cosine matrix which is used to build a tree for the PageRank algorithm:

**TABLE 1: COSINE SIMILARITY MATRIX**

<table>
<thead>
<tr>
<th></th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
<th>S6</th>
<th>S7</th>
<th>S8</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>1.00</td>
<td>0.15</td>
<td>0.74</td>
<td>0.00</td>
<td>0.85</td>
<td>0.00</td>
<td>0.20</td>
<td>0.25</td>
</tr>
<tr>
<td>S2</td>
<td>0.15</td>
<td>1.00</td>
<td>0.15</td>
<td>0.71</td>
<td>0.26</td>
<td>0.59</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>S3</td>
<td>0.74</td>
<td>0.15</td>
<td>1.00</td>
<td>0.00</td>
<td>0.57</td>
<td>0.00</td>
<td>0.20</td>
<td>0.57</td>
</tr>
<tr>
<td>S4</td>
<td>0.00</td>
<td>0.71</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.80</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>S5</td>
<td>0.85</td>
<td>0.26</td>
<td>0.57</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.34</td>
<td>1.00</td>
</tr>
<tr>
<td>S6</td>
<td>0.00</td>
<td>0.59</td>
<td>0.00</td>
<td>0.80</td>
<td>0.00</td>
<td>1.00</td>
<td>0.59</td>
<td>0.80</td>
</tr>
<tr>
<td>S7</td>
<td>0.20</td>
<td>1.00</td>
<td>0.20</td>
<td>0.00</td>
<td>0.34</td>
<td>0.59</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>S8</td>
<td>0.25</td>
<td>0.00</td>
<td>0.57</td>
<td>0.00</td>
<td>1.00</td>
<td>0.80</td>
<td>0.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

The generated encapsulated text:

> Model predicts declining deaths, but vaccine hesitancy could hurt daily Covid-19 deaths in the US. Should continue to decline in the coming months, though that could change if vaccine hesitancy rises, Murray's IHME said Friday. So far, no vaccines are authorized for people younger than 16, but a study published in the medical journal JAMA Network Open on Friday showed that quickly identifying and contact tracing children to identify "silent infections" of Covid-19, where the disease is either presymptomatic or asymptomatic, combined with vaccination of 40-60% adults could significantly reduce the amount of disease.

8.3 **Approach 3 – Dual bi-directional LSTM**

From the previous 2 approaches we can infer that no new text is generated but simply sentences are chosen based on the assigned weights and ranks. The sentences are then displayed as output. However, these sentences do not capture the essence of the input text and sound vaguely odd when read by the user. This is where the developed system beats all the existing systems. This is due to use of three LSTM layers out of which two are bidirectional and all the layers are interlinked to each other. The input text is given to LSTM layer 1 which generates intermediate return sequences along with the return state. This is then fed to LSTM layer 2 which generates a new set of sequences and state and is then parsed by the LSTM layer 3. The LSTM layer 3’s output is then
used as input by decoder that uses SoftMax activation function along with CAC and generates new text that captures the holistic meaning of the entire input text.

```python
# Get the input to Encoder
ips = Input(shape=(max_len_text,))
bedd = Embedding(voc_size, lDim, trainable=True)(ips)

# 1 LSTM
en_1l2 = LSTM(lDim, return_sequences=True, return_state=True)(en_1l1, initial_state=enc_emb)

# 2 LSTM
en_1l2 = LSTM(lDim, return_sequences=True, return_state=True)(en_1l1, initial_state=en_1l2(enc_emb))

# 3 LSTM
en_1l3 = LSTM(lDim, return_sequences=True, return_state=True)(en_1l2(en_1l1), initial_state=en_1l3(en_1l2))

def configure_decoder():
    dc_1l = Input(shape=(None,))
dc_enc = Embedding(voc_size, lDim, trainable=True)(dc_1l)
dc = Embedding((None, dc_enc)

# The initial state of the LSTM as encoder states
dr_1l = LSTM(lDim, return_sequences=True, return_state=True)(dr_1l, initial_state=enc_1l)

# Context Aware Consciousness
intermediate_sequence = 3
cac = AttentionLayer(name='attention_layer')(enc_1l, initial_state=enc_1l)

def Concatenation(name='concat_layer')(dr_1l, cac)

# Dense Layer
dense = Dense(voc_size, activation='softmax')(fc)
dr_1l = Dense(dense)

# Define the model
model = Model([ips, dc, dr_1l])
model.summary()

Model: "model"

Layer (type)                  Output Shape         Param #     Connected to
=================================================================
input_1 (InputLayer)         [(None, 60)]         0            
embedding (Embedding)        [(None, 80, 500)]    2578500      input_1[0][0]

lstm (LSTM)                  [(None, 500), (N 2002000) 2578500 + embedding[0][0]

lstm_1 (LSTM)                [(None, 80, 500), (N 2002000) 2578500 + lstm[0][0]

The generated encapsulated output:

The US may have more vaccines than people who want them by mld-May. Here's why that's a big problem.
To understand how the model performs on both the datasets, analysis of distribution of text length is performed and a graph is plotted using matplot. Y-axis represents the total number of statements X-axis represents the number of words.

Figure 9: Distribution of text length for Amazon Reviews Dataset

As seen in figure 9, the average length of reviews ranges from 200 – 400 words per review.

Figure 10: Distribution of text length for CNN News Dataset

As seen in figure 10, the average length of news articles ranges from 1000 – 2500 words per article.
Sample outputs generated for Amazon Reviews dataset with the help of approach 3:

Original Summary: bars tasty really taste like brownie certainly chocolatey yummy good amount protein good afternoon snack trying gain muscle mass best thing workout contain almost carbs know proper
Generated Encapsulation: pretty good for protein bar

Original Summary: really like taste powder great protein shake especially banana blended issue shelf life product container lasts long time container purchased amazon expired less month purchase date
Generated Encapsulation: good in smoothies but why the short shelf life

Original Summary: self respecting cat picky eater never met cat likes whiskas choice cut foods mainly gravy sometimes meat magnificent wonderful likes beef gravy tuna truth told dry crunchie really
Generated Encapsulation: good cat food without smelly cans

Original Summary: better mayo one market also one made without sugar complete miracle way mayonnaise also great spices tangy taste perfect would expect lemon mayonnaise whole foods near home carried
Generated Encapsulation: best mayo on the market

Original Summary: shipping fine one packs see costo placed box shipment good deal around dollar bottle drink made splenda plenty sweet almost taste like regular juice drink one better tasting diet
Generated Encapsulation: best tasting splenda drink with no after taste

Original Summary: diet fresh easy neighborhood grocery looking possible diet foods wanted things tasty low fat low calories come home dozen items discovered farm pickled crispy asparagus
Generated Encapsulation: my idea of good diet food

Original Summary: ordered line received days tea excellent price right
Generated Encapsulation: good tea

Original Summary: organic market bought two bottles cost three dollars fell love instantly drink delicious taste yerba mate tea juice like slightly bitter chocolate honey tea sensitive sucrosolve
Generated Encapsulation: best energy shot have ever tasted

Original Summary: bought years grocery store never carry anymore ended getting great price bulk package highly recommended kool aid type drink
Generated Encapsulation: one of the best drinks moves out there

Original Summary: great brownies happen gluten free everyone loves including wheat eaters celiacs disease cannot eat gluten get ill never gotten ill brownies means gluten free safe gluten free diets
Generated Encapsulation: great brownies that just happen to be gluten free

Original Summary: local eat regularly use kinds foods love hot without flavor like sauces burn taste burn awesome flavor surprised find amazon would highly recommend anyone likes good hot sauce
Generated Encapsulation: love this stuff

Original Summary: good even store brand oatmeal requiring preparation still mccormick good gets instant oatmeal even better organic natural brands tried varieties mccormick variety pack taste good microwave
Generated Encapsulation: best of the instant

Original Summary: spicy chips let great flavor left breath afterwards first time ate night even could taste mouth next morning careful eat especially people later day
Generated Encapsulation: tasty but make sure you have gum

Original Summary: quaker oatmeal cookies taste make easy snack work individually wrapped put bag cookie soft nice bits nuts chocolate little dry perfect milk brainer good cookie individually wrapped l
Generated Encapsulation: good cookie for on the go

Original Summary: popchips really great snack alternative regular chips full replacement far concerned taste texture great jalapeno ones hate greasy fried chips often switch baked version chips another
Generated Encapsulation: great alternative to regular chips

Original Summary:
Sample outputs generated from CNN news Dataset:

Original title: some start selling rights shot many elephants year country announced week spokesman zimbabwe parks wildlife management authority told cnn thursday declining tourism revenue using coronavirus pandemic among main
Generated Encapsulation: decision to allow elephant hunting was appalling read more

Original title: wrote first time seen was smile since wife died dan peterson lost beloved wife march fall deep depression six months later chance encounter store day going day self pity feeling sorry told cnn changed
Generated Encapsulation: how you can positively impact life turn wood said

Original title: granted temporary residency allowing right work began four year long process gain full citizenship navigating complexities financial costs american immigration bureaucracy arriving new country thought activism
Generated Encapsulation: trump emphasis on fighting for ordinary americans read more

Original title: my total symptoms get cnn health weekly newsletter sign get results singjay google every tuesday cnn health team parent keep watch symptoms reach immediately health care provider emergency room one last important
Generated Encapsulation: close contact with your pediatrician and your pediatrician

Original title: responsibility enabling government said report published monday april july people slaughtered mainly ethnic minority also message foreign affairs minister today key stop getting two countries closer french presid
Generated Encapsulation: france role in connection with the genocide read more

Original title: why allows seem said share back see connection storing story creating shared vision together says managing loneliness post pandemic world simply take time effort people talk makes realize going try start said
Generated Encapsulation: that as we emerge from pandemic we emerge together

Original title: allowed observe understand catherine mayer monarchy ultimate exclusive club sovereign born reigns subjects citizens hardly obvious qualifications performing key role head state queen nonetheless largely succeed
Generated Encapsulation: to avoid restrictions it must seen proportionate read more

Original title: weeks world health organization warned virus swept hitting several corners globe million now cases recorded last week single week since pandemic began director general tedros adhanom ghebreyesus said news
Generated Encapsulation: these deaths is tragedy for families communities and nations

Original title: deadly tesla crash texas raised questions tesla autopilot feature consumer reports released video showing someone might trick autopilot system operate without driver
Generated Encapsulation: autopilot system to operate without driver source cnn business

As seen from the output above, the original titles and summaries may not depict the exact information that the text carries. The encapsulation that is generated using the deep learning model captures the holistic meaning of the text. Some outputs generated with this approach may not be precise or some might be blank. This is due to the loss encountered during the model training and validation phases.
8.4 Model Training and Validation Loss

The observations from experiments provide a deep insight into the results. There is a notable performance difference between the 3 approaches. The first methodology generates moderate results with a marginal score of 67 percent using Word-frequency as a measure, the second methodology performed better with the Text Ranking algorithm with a good accuracy of 78 percent, and the third method has a 93 percent accuracy using the proposed LSTM network.

The training and test phase losses for each of the models decreases gradually as we move from Epoch 1 to Epoch 10. An insight into losses for each of these models helps better understand why the approach 3 has high accuracy. For approach 1 and 2, a significant change in validation loss even after 10 epochs cannot be seen.

TABLE 2: Training and Validation Loss per Epoch for Scoring Sentences (Approach 1)

<table>
<thead>
<tr>
<th>EPOCHS</th>
<th>TRAINING LOSS</th>
<th>VALIDATION LOSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/10</td>
<td>8.933</td>
<td>8.900</td>
</tr>
<tr>
<td>2/10</td>
<td>7.297</td>
<td>7.013</td>
</tr>
<tr>
<td>3/10</td>
<td>6.662</td>
<td>6.190</td>
</tr>
<tr>
<td>4/10</td>
<td>6.288</td>
<td>6.121</td>
</tr>
<tr>
<td>5/10</td>
<td>6.088</td>
<td>5.759</td>
</tr>
<tr>
<td>6/10</td>
<td>5.407</td>
<td>5.366</td>
</tr>
<tr>
<td>7/10</td>
<td>5.328</td>
<td>5.049</td>
</tr>
<tr>
<td>8/10</td>
<td>5.076</td>
<td>5.003</td>
</tr>
<tr>
<td>9/10</td>
<td>4.891</td>
<td>4.558</td>
</tr>
<tr>
<td>10/10</td>
<td>4.322</td>
<td>4.219</td>
</tr>
</tbody>
</table>
TABLE 3: Training and Validation Loss per Epoch for Text Ranking (Approach 2)

<table>
<thead>
<tr>
<th>EPOCHS</th>
<th>TRAINING LOSS</th>
<th>VALIDATION LOSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/10</td>
<td>8.323</td>
<td>8.157</td>
</tr>
<tr>
<td>2/10</td>
<td>8.091</td>
<td>8.073</td>
</tr>
<tr>
<td>3/10</td>
<td>7.866</td>
<td>7.739</td>
</tr>
<tr>
<td>4/10</td>
<td>7.600</td>
<td>7.577</td>
</tr>
<tr>
<td>5/10</td>
<td>4.945</td>
<td>4.825</td>
</tr>
<tr>
<td>6/10</td>
<td>4.507</td>
<td>4.447</td>
</tr>
<tr>
<td>7/10</td>
<td>4.226</td>
<td>4.140</td>
</tr>
<tr>
<td>8/10</td>
<td>3.505</td>
<td>3.428</td>
</tr>
<tr>
<td>9/10</td>
<td>3.110</td>
<td>3.107</td>
</tr>
<tr>
<td>10/10</td>
<td>3.086</td>
<td>3.079</td>
</tr>
</tbody>
</table>

TABLE 4: Training and Validation Loss per Epoch for Dual LSTM (Approach 3)

<table>
<thead>
<tr>
<th>EPOCHS</th>
<th>TRAINING LOSS</th>
<th>VALIDATION LOSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/10</td>
<td>6.913</td>
<td>6.125</td>
</tr>
<tr>
<td>2/10</td>
<td>6.856</td>
<td>5.751</td>
</tr>
<tr>
<td>3/10</td>
<td>4.909</td>
<td>3.297</td>
</tr>
<tr>
<td>4/10</td>
<td>3.485</td>
<td>2.741</td>
</tr>
<tr>
<td>5/10</td>
<td>3.143</td>
<td>2.729</td>
</tr>
<tr>
<td>6/10</td>
<td>3.077</td>
<td>2.620</td>
</tr>
<tr>
<td>7/10</td>
<td>2.971</td>
<td>2.594</td>
</tr>
<tr>
<td>8/10</td>
<td>2.950</td>
<td>2.524</td>
</tr>
<tr>
<td>9/10</td>
<td>2.854</td>
<td>2.405</td>
</tr>
<tr>
<td>10/10</td>
<td>2.831</td>
<td>2.463</td>
</tr>
</tbody>
</table>
8.5 Run-Time Analysis

The model training sequence is the most time consuming step as the model is being trained to read unknown sequences of text and generate a precise output. For the first approach that involves scoring sentences based on the word frequency, the time required for model training and evaluation is the least. This is why it has the least accurate output. For the covid news snippet with over 1250 words, end to end run with the approach 1 takes approximately 25 seconds to generate output. Approach 2 which uses text ranking, performs better in terms of generating a precise output but its model training and evaluation requires more time as compared to the first approach. For the news snippet above, end to end run with the approach 2 takes approximately 30 seconds to generate output text.

For approach 3, in the very first training phase, the developed system requires the largest amount of time to train and evaluate the model. In this first phase, the model training and evaluation takes approximately 42 seconds on a small Amazon reviews dataset with over 1300 records. From the second phase onwards, the run-time decreases as the model gets trained on more new data. This model training and evaluation takes approximately 16 seconds to complete in the second run. The larger Amazon dataset is over 300MB in size and consists of 100,000 reviews. The model training and validation on this dataset takes approximately 22 minutes. Although the time required to train the model is more, it generates a precise output. For the covid news snippet above, the approach 3 takes approximately 2 milliseconds to generate the encapsulated text. The table 2 displays the runtime analysis information for all the approaches.


TABLE 5: Run-time Analysis

<table>
<thead>
<tr>
<th>Input Data</th>
<th>Approach</th>
<th>Total Run-time for Model Training and Evaluation</th>
<th>Time to generate Output (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scoring Sentences</td>
<td>Scoring Sentences based on the Word-Frequency</td>
<td>0:00:24.746911</td>
<td>24.746911</td>
</tr>
<tr>
<td>News Snippet</td>
<td>Universal Sentence Encoder for Text Ranking</td>
<td>0:00:30.07571</td>
<td>30.07571</td>
</tr>
<tr>
<td>News Snippet</td>
<td>Dual bi-directional LSTM</td>
<td>0:00:30.07571</td>
<td>2.097957</td>
</tr>
<tr>
<td>Amazon Reviews</td>
<td>Dual bi-directional LSTM</td>
<td>0:00:42.349841</td>
<td>1.086803</td>
</tr>
<tr>
<td>Dataset 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Amazon Reviews</td>
<td>Dual bi-directional LSTM</td>
<td>0:22:46.944088</td>
<td>12.744386</td>
</tr>
<tr>
<td>Dataset 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CNN News Dataset</td>
<td>Dual bi-directional LSTM</td>
<td>0:03:48.934982</td>
<td>1.097957</td>
</tr>
</tbody>
</table>

8.6 Comparison of the Results

All the above approaches performed differently with different algorithms used in each implementation. The accuracy is a good measure for evaluating the models, along with the F1 scores that can be used for comparison. The comparisons of accuracy and F1 scores from the results of each approach, led to insightful conclusions. Scoring Sentences based on the Word-Frequency in approach 1 has a low performance and accuracy of 67%. In the second approach, Universal Sentence Embeddings have a slightly higher performance as compared to approach 1.
with an accuracy of 78%. The LSTM model in approach 3 performed the best as it closely resembles a human summary and has the highest accuracy of 93% with high F1 scores.

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scoring Sentences based on the Word-Frequency</td>
<td>67% (+/- 0.537)</td>
<td>65%</td>
<td>61%</td>
<td>66%</td>
</tr>
<tr>
<td>Universal Sentence Encoder for Text Ranking</td>
<td>78% (+/- 0.029)</td>
<td>77%</td>
<td>75%</td>
<td>76%</td>
</tr>
<tr>
<td>Dual bi-directional LSTM</td>
<td>93% (+/- 1.046)</td>
<td>92%</td>
<td>91%</td>
<td>93%</td>
</tr>
</tbody>
</table>

The following remarks can be drawn from the findings:

Remark 1: The first approach simply scores all the sentences and selects the highest weighted sentences to display as summarized output. We can infer from Table 2 that since the training and validation phase losses are high, this model has an average precision score of 65%. The holistic meaning of the text might not be captured by this method. Even though it has a good F1 score, the recall is pretty poor for this approach.

Remark 2: The second approach has a good accuracy of 78% due to moderate training and test phase losses during the 10 Epochs. This is mainly because of the Text Ranking grid matrix. Text Ranking uses cosine function to determine the similarity of 2 sentences to each other. This cosine
linear similarity matrix is then used to build a tree. The PageRank rating equation is then added to the tree in order to determine rankings for each statement. The summaries generated by this approach has low recall but high precision and F1 scores of 77% and 76% respectively.

Remark 3: The proposed approach as demonstrated in the experiments, has the highest accuracy out of the 3 approaches. This is because it has the lowest training and validation phase losses during the 10 epochs. This approach generates sentences based on the natural language understanding with the help of SoftMax and CAC. This approach more closely resembles a human approach rather than a machine generated approach. This is why it has a high precision of 92% and F1 score of 93%. Even though the recall is less, the overall accuracy of the approach turns out to be greater than 93%.
IX. CONCLUSION AND FUTURE WORK

The modern age of innovation has begun as a result of the rapid advancement in the realms of deep learning. Most of these innovations are automated and require little to no human intervention in generating precise output. Text encapsulation means capturing the meaning of the sequence of text rather than just summarizing the text. Existing text summarization systems like approaches 1 and 2 simply remove pronouns, some random words and summarize the articles based on sentence weights. These summarizations can be completely different in meaning when compared with the original article. The developed model captures the exact meaning of the entire text by parsing the input data through multiple LSTM layers and generates its own encapsulated text. Since this model has high precision of 92% and F1 score of 93%, the generated text closely resembles a human generated one or in some cases even better than the human generated ones.

In a world where pandemic is prevalent for a long time and news with mis represented headings lure people into disappointment, it is more important than ever to have a robust text encapsulation system. The developed model can be further refined by using a large dataset in combination with a quadri-bidirectional LSTM network. By making use of the beam scanning technique, this model can be further improved, loss can be reduced and higher output accuracy can be attained. The developed system’s applications can be extended to academia as well where it can be used by budding researchers to encapsulate brief articles into some well-structured short descriptions or titles. This research project provides an overview on existing systems and proposes an approach to automate the text encapsulation process. The series of tests performed on the developed model using heterogeneous datasets make it robust, consistent and highly efficient as compared to existing systems.
REFERENCES


