**Abstractive Text Summarization**

***A Report submitted***

***in partial fulfillment for the Degree of***

***Bachelor of Technology***

***in***

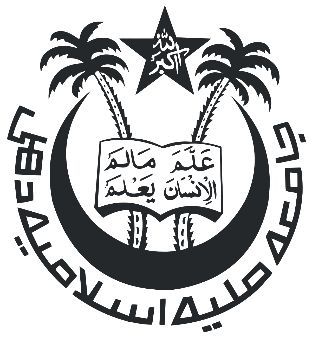
***Computer Engineering***

***by***

***Habiburrahman 17BCS071***

***Khan Mohd Arquam 17BCS008***

***Saman Rashid 17BCS094***



***Department of Computer Engineering***

***Faculty Engineering & Technology,***

***Jamia Millia Islamia New Delhi –110025***

# **ABSTRACT**

Neural sequence-to-sequence models have provided a viable new approach for abstractive text summarization (meaning they are not restricted to simply selecting and rearranging passages from the original text). However, these models have two shortcomings: they are liable to reproduce factual details inaccurately, and they tend to repeat themselves. In this work we propose a novel architecture that augments the standard sequence-to-sequence attentional model in two orthogonal ways. First, we use a hybrid pointer-generator network that can copy words from the source text via pointing, which aids accurate reproduction of information, while retaining the ability to produce novel words through the generator. Second, we use coverage to keep track of what has been summarized, which discourages repetition. We apply our model to the CNN / Daily Mail summarization task, outperforming the current abstractive state-of-the-art by at least 2 ROUGE points.

Contents

[**ABSTRACT** 2](#_Toc75953507)

[**1.1** **Background** 4](#_Toc75953508)

[**1.1.1 What is text Summarization** 4](#_Toc75953509)

[**1.1.2 Extractive Summarization** 4](#_Toc75953510)

[**1.1.3 Abstractive Summarization** 4](#_Toc75953511)

[**2. Literature Review** 5](#_Toc75953512)

[**3. PROBLEM STATEMENT** 10](#_Toc75953513)

[**3.1 Inaccurate actual Details** 10](#_Toc75953514)

[**3.2 Repetition of Sentences** 10](#_Toc75953515)

[**4. PROPOSED MODEL** 11](#_Toc75953516)

[**4.1 Pointer-generator network** 11](#_Toc75953517)

[**4.2 Coverage mechanism** 12](#_Toc75953518)

[**5. IMPEMENTATION** 14](#_Toc75953519)

[**5.1 Dataset** 14](#_Toc75953520)

[**5.1.2 Dataset Preparation** 14](#_Toc75953521)

[**5.2 Hyper Parameters Values** 15](#_Toc75953522)

[**5.3 Tensorflow Model** 15](#_Toc75953523)

[**5.4 Experimental Results:** 16](#_Toc75953524)

[**6.1 Conclusion** 20](#_Toc75953525)

[**6.2 Suggestions for future work** 20](#_Toc75953526)

[**6.3 References** 20](#_Toc75953527)

**1.INTRODUCTION**

# **Background**

## **1.1.1 What is text Summarization**

The task of condensing a piece of text/article to a more concise version containing the important and requisite information from the original text.

There are two approaches to summarization:

* Extractive Summarization
* Abstractive Summarization

## **1.1.2 Extractive Summarization**

Extractive methods generate the summary exclusively from the original text (often whole sentences) directly copied from the original text/article, the extractive method is simpler, since copying large amount of text from the source text often results in accurate grammar and also less misinformation.

**Summary**

Sentence 2

Sentence 4

**Text**

Sentence 1

Sentence 2

Sentence 3

Sentence 4

Extractive Summarizer

**An illustration of the abstractive summarization**

## **1.1.3 Abstractive Summarization**

The abstractive methods of summarization is used to generate new words and phrases that are not present in the original document. Hence, this is considered much similar to a human written summary.

Much complex abilities that are very important to achieve human-like summarization, such as generalization, the addition of context knowledge, or paraphrasing, are possible only in the abstractive method.

Summary

New Sentences

Text

Sentence 1

Sentence 2

Sentence 3

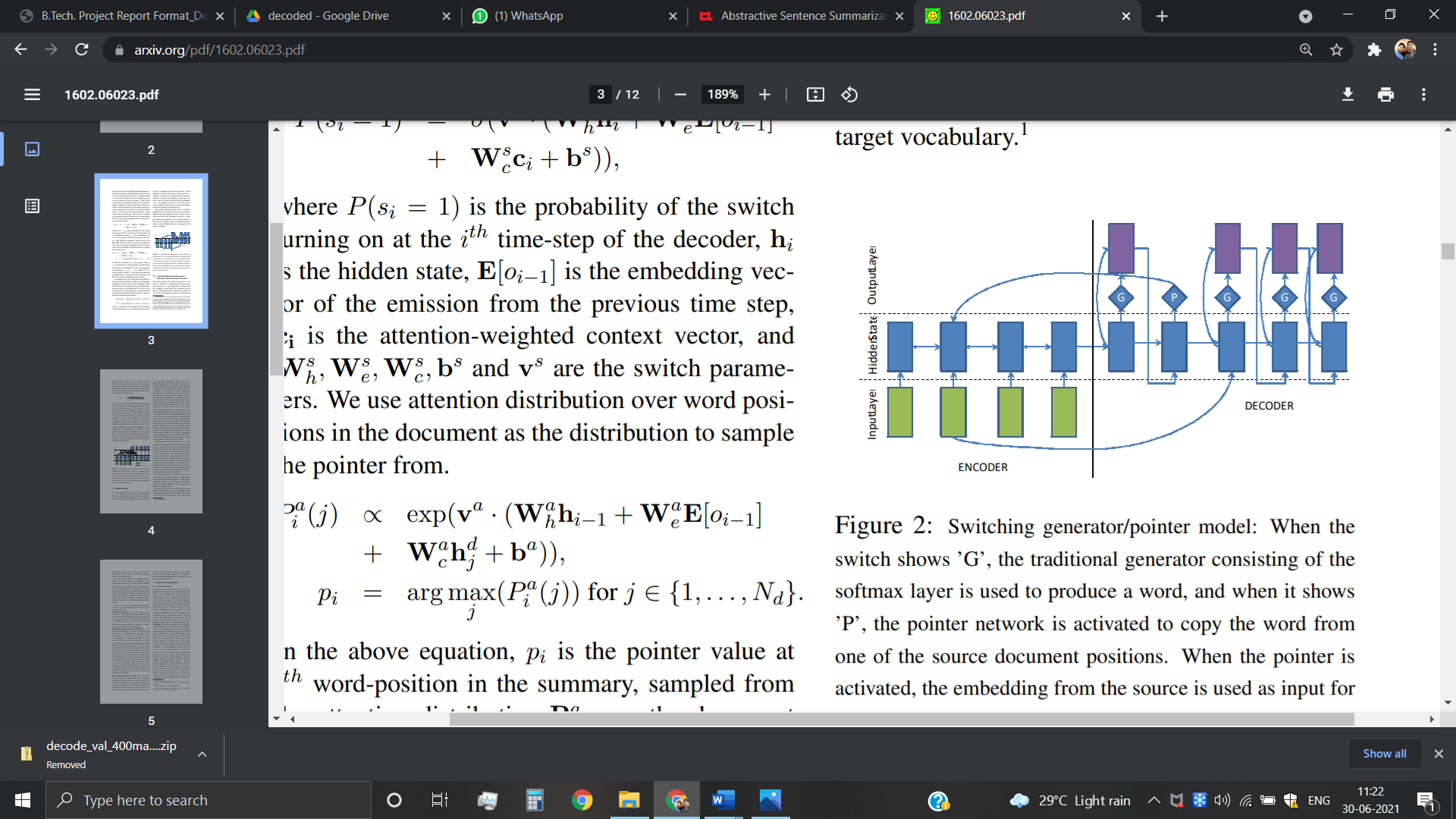
Sentence 4

Abstractive Summarizer

**An illustration of the abstractive summarization**

# **2. Literature Review**

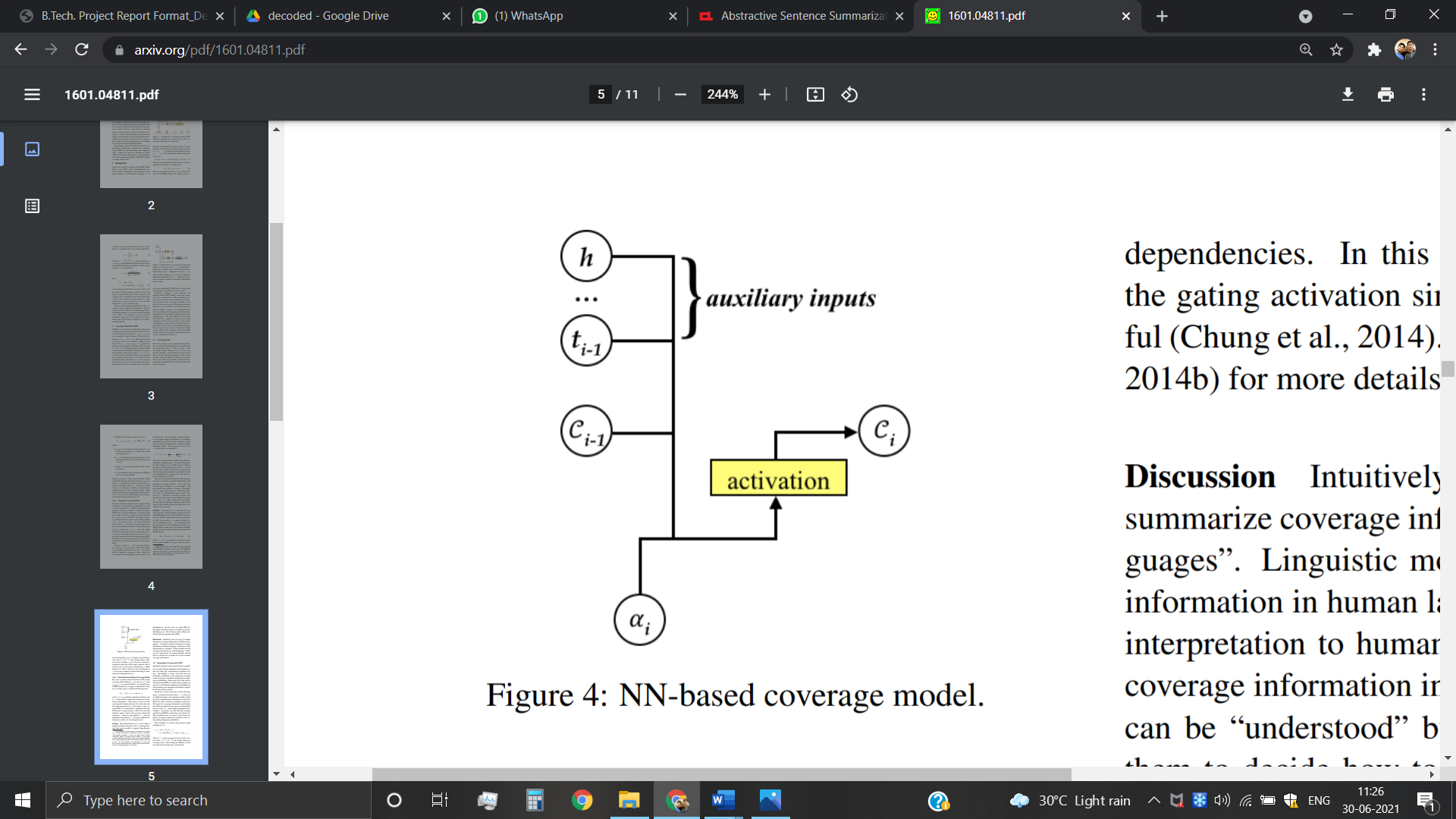
Since the abstractive method of summarization is quite complex and difficult to achieve, the majority of the past work has been based on the extractive method of summarization (Kupiec et al., 1995; Paice, 1990; Saggion and Poibeau, 2013). But however, the recent success of sequence-to-sequence models (Sutskever el at 2014 ) , wherein recurrent neural networks (RNNs) have the ability to both read and generate summaries on their own, has made the seemingly impossible task of abstractive summarization look possible (Chopra et al., 2016; Nallapati et al., 2016; Rush et al., 2015; Zeng et al., 2016). Even though these works are very good and should be acknowledged, they have their own flaws such as they inaccurately reproduce factual information, and are also uncapable of dealing with words not present in our vocabulary, and they often repeat words/phrases.



**Switching generator/pointer model as Proposed by Nallapati et al, 2016**

In this project we propose a much newer model that tries to solve these three major problems in the context of multi-sentence summaries. However, most recent abstractive work has been focused on generating headlines of news articles (summarizing one or two sentences to a single sentence), but here we take a different path which will be more useful and is a requirement today. Hence we work our proposal to the CNN/ Daily Mail dataset (Hermann et al., 2015; Nallapati et al., 2016), which contains news articles along with reference summaries having multiple sentences.

The proposed pointer-generator model enables extraction of words from the original document (Vinyals et al., 2015), this helps in managing the problem of Out of Vocabulary words and also helps improve accuracy, at the same time it also keeps on its task of generating new and novel words. This network, can be seen as a combination of extractive and abstractive methods, hence it can be considered similar to Gu et al.’s (2016) CopyNet and Miao and Blunsom’s (2016) Forced-Attention Sentence Compression, that were applied to short-text summarization. A coverage vector (Tu et al., 2016) from Neural Machine Translation is also used, which will enable us to manage coverage of the original text. Hence, we show that coverage will reduce repetition.



**Neural Network based Coverage model from Tu et al., 2016**

Firstly, as per the work done on Neural Sequence to Sequence attention models, these models have shown results that are quite promising in context of Abstractive Text Summarization. However, there are many issues that pertain. The generated summaries are often meaningless, hence many techniques have been tried to solve these issues.

The reinforcement learning based training procedures using intra-Attention that improves the model’s accuracy remarkably has been explored as well. The problems that plagued the area in detail, and the possible ways to improve those areas were also analyzed in previous works. Also a novel architecture was proposed to solve the issues and problems in summary generation of longer text which were difficult to be captured with currently ongoing used models. With their deep learning approaches, they succeeded to a great extent and finally concluded that Deep Learning approach are quite promising and would be helpful in the near future to solve the abstractive text summarization issues in the near future.

However, the tough task of scalability and generalization process of multi- sentence summarization is plagued by problems with the evaluation metrics(ROUGE, METEOR) and lack of dataset availability.

Hence it was finally concluded that Deep Learning approach are much relevant and would be very helpful in further research in this domain and to tackle the challenge of abstractive text summarization.

Talking about other works, the usage of Sequence-to-Sequence model was shown and developed by Tian Shi et al. According to them, neural sequence-to-sequence model has become quite renowned in the past years. To handle future challenges in this domain of text summarization, many interesting techniques have been developed. This will improve the sequence-to-sequence models. Also, they believe that the majority of the techniques vary among the three categories: network structure, parameter inference and decoding/generation. At times, efficiency of a model becomes a concern during the training phase. In their paper publication, a detailed literature as well as technical survey on various sequence-to-sequence models for abstractive text summarization was presented.

Many of their models were primarily used and proposed for modelling of the language and task generation, such as machine translation, and were applied later to abstractive text summarization.

An open source library was also developed, named as Neural Abstractive Text Summarizer (NATS) toolkit, to enable the task of abstractive text summarization, during their survey. On the CNN/Daily Mail dataset a number of examinations were conducted to evaluate its effects of various different neural network components. Finally, Two models implemented in NATS on two novel contributed datasets, such as, Newsroom and Bytecup were compared and confirmed. Hence, their effectiveness in their work was proved. Therefore, it was concluded that this application was quite successful for sequence-to-sequence models. Hence, neural abstractive text summarization has been a quite popular research area and is in demand from both the industry as well as the academics.

Further, going ahead to see something different, is the use of Feed-Forward network by Lu Yang, This focus of this research is to use the feed-forward neural network with an attention-based encoder in order to meet the problems of abstractive text summarization. They analyzed the attentive recurrent neural network and recurrent neural network encoder-decoder to check its effectiveness and power. These discussed models were mainly developed for solving tasks, such as summarization of newspaper publication and translation of machine context. They made assessment on model’s advantages using ROUGE and results of visual inspection by modifying and furthering these models to the issue of product review summarization. In the end, they discovered the outcomes to be favourable.

**Table:- A COMPARATIVE STUDY ON Abstractive Summarization**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Year** | **References** | **Highlight** | **Model used** | **Dataset** | | **Outcome** |
| 2015 | Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sen- tence summarization. In *Empirical Methods in Nat- ural Language Processing*. | Reinforcement learning based training procedures using intra-Attention. | Neural Sequence to Sequence Model | the standard sentence summarization evaluation set is associated with the DUC-2003 and DUC2004 | | they have presented a neural attention-based model for abstractive summarization, based on recent developments in neural machine translation. they combine this probabilistic model with a generation algorithm which produces accurate abstractive summaries. |
| 2016 | Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. 2016. Neural headline generation on abstract meaning representation. In Empirical Methods in Natural Language Processing. | Neural network-based encoder-decoder models are among recent attractive methodologies for tackling natural language generation tasks | AMR , 2 Attention-Based AMR Encoder | data of the abstractive headline generation task described in Rush et al. (2015). | | The experimental results of headline generation benchmark data showed that our attention-based AMR encoder-decoder model successfully improved standard automatic evaluation measures of headline generation tasks, ROUGE-1, ROUGE-2, and ROUGEL |
| 2016 | Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with at- tentive recurrent neural networks. In *North Amer- ican Chapter of the Association for Computational Linguistics*. | introduce a conditional recurrent neural network (RNN) which generates a summary of an input sentence. The conditioning is provided by a novel convolutional attention-based encoder which ensures that the decoder focuses on the appropriate input words at each step of generation. | Attentive Recurrent Architecture , Recurrent Decoder , Attentive Encoder | the Gigaword corpus (Graff et al., 2003; Napoles et al., 2012) | | extend the state-of-the-art model for abstractive sentence summarization (Rush et al., 2015) to a recurrent neural network architecture. This model is a simplified version of the encoder-decoder framework for machine translation (Bahdanau et al., 2014). |
| 2016 | Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, C¸ aglar Gulc¸ehre, and Bing Xiang. 2016. Abstrac- tive text summarization using sequence-to-sequence RNNs and beyond. In *Computational Natural Lan- guage Learning*. | Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond | Encoder-Decoder RNN with Attention and Large Vocabulary Trick , Capturing Keywords using Feature-rich Encoder | | Gigaword Corpus | propose several novel models that address critical problems in summarization that are not adequately modeled by the basic architecture, such as modeling key-words, capturing the hierarchy of sentence-toword structure, and emitting words that are rare or unseen at training time |

# **3. PROBLEM STATEMENT**

## **3.1 Inaccurate actual Details**

The summaries sometimes **emulate factual details incorrectly**. This is most common for infrequent or words beyond vocabulary.

It is too difficult to copy a word ‘w’ from the source text with the sequence-to-sequence-with attention model. Essentially, the original word must somehow be recovered by the network after the information has passed through several layers of computation (including mapping w to its word Embedding).

This is especially true if w has a poor word embedding (i.e. it is clustered with completely unrelated words) because it is a sparse/rare word that develops infrequently during training and therefore, then w is, from the understanding of the network, identical from many other words and hence impossible to duplicate.

The network may still have difficulty reproducing the word even if w has a good word embedding. For example, RNN summarization systems often mistake and erroneously substitute a name with another name (e.g. Mary → Katie) or a city with another city (e.g. Kolkata → Chennai). This is because the word embeddings for e.g. names of women or names of Indian cities tend to cluster together, which may cause confusion when attempting to reproduce the original word.

In short, this seems like an unnecessarily difficult way to perform a simple operation – copying – that is a fundamental operation in summarization

## **3.2 Repetition of Sentences**

The summaries sometimes **repeat a said statement** (e.g. *Austria beat Austria beat Austria beat…*)

These problems are not rare for RNNs in general. It is difficult to explain *why* the network exhibits any particular behavior, a well-known characteristic of deep learning. For those who are interested, I offer the following conjectures-

Repetition may be caused by the decoder’s *over-reliance on the decoder input (i.e. previous summary word)*, rather than storing longer-term information in the decoder state. A single repeated word commonly triggers an endless repetitive cycle, prove this. For example, a single substitution error *Austria beat****Austria*** leads to the catastrophic *Austria beat Austria beat Austria beat…*, and not the less-wrong *Austria beat Austria 2-0*.

# **4. PROPOSED MODEL**

In this section we describe

(1) pointer generator model

(2) coverage mechanism.

# **4.1 Pointer-generator network**

Our baseline model is similar to that of Nallapati et al. (2016), and is depicted in Figure 2. The tokens of the article wi are fed one-by-one into the encoder (a single-layer bidirectional LSTM), producing a sequence of encoder hidden states hi . On each step *t*, the decoder (a single-layer unidirectional LSTM) receives the word embedding of the previous word (while training, this is the previous word of the reference summary; at test time it is the previous word emitted by the decoder), and has decoder state st . The attention distribution at is calculated as in Bahdanau et al. (2015):

**eti = vT tanh(Whhi +Wsst +battn) (1)**

**at= softmax(et ) (2)**

where v, Wh, Ws and battn are learnable parameters. The attention distribution can be viewed as

a probability distribution over the source words, that tells the decoder where to look to produce the next word. Next, the attention distribution is used to produce a weighted sum of the encoder hidden states, known as the context vector h\*t:

**ht∗ = ∑i at hi (3)**

The context vector, which can be seen as a fixed size representation of what has been read from the source for this step, is concatenated with the decoder state st and fed through two linear layers to produce the vocabulary distribution Pvocab:

***P*vocab = softmax(*V,*(*V* [*st , ht∗*] + *b*) + b,) (4)**

where V, V, , b and b, are learnable parameters. Pvocab is a probability distribution over all words in the vocabulary, and provides us with our final distribution from which to predict words w:

**P(w) = Pvocab(w) (5)**

During training, the loss for timestep t is the negative log likelihood of the target word w\*t for that timestep:

**Losst = −logP(w\*t ) (6)**

and the overall loss for the whole sequence is:

**loss = 1 ( ∑t = 0 T losst ) / T**

# **4.2 Coverage mechanism**

Repetition is a common problem for sequence to sequence models (Tu et al., 2016; Mi et al., 2016; Sankaran et al., 2016; Suzuki and Nagata, 2016), and is especially pronounced when generating multi-sentence text (see Figure 1). We adapt the coverage model of Tu et al. (2016) to solve the problem. In our coverage model, we maintain a coverage vector ct , which is the sum of attention distributions over all previous decoder timesteps:

**Ct= ∑t-1tt=0 at~ (10)**

Intuitively, ct is a (unnormalized) distribution over the source document words that represents the degree of coverage that those words have received from the attention mechanism so far. Note that c0 is a zero vector, because on the first timestep, none of the source document has been covered. The coverage vector is used as extra input to the attention mechanism, changing equation (1) to:

**eti = vT tanh(Whhi +Wsst + wc ct+battn) (11)**

where wc is a learnable parameter vector of same length as v. This ensures that the attention mechanism’s current decision (choosing where to attend next) is informed by a reminder of its previous decisions (summarized in ct ). This should make it easier for the attention mechanism to avoid repeatedly attending to the same locations, and thus avoid generating repetitive text. We find it necessary (see section 5) to additionally define a coverage loss to penalize repeatedly attending to the same locations:

**covlosst = ∑imin(ati, cit) (12)**

Note that the coverage loss is bounded; in particular

**Covlosst ≤ ∑iait = 1. Equation (12)**

differs from the coverage loss used in Machine Translation. In MT, we assume that there should be a roughly one to one translation ratio; accordingly the final coverage vector is penalized if it is more or less than 1.

Our loss function is more flexible: because summarization should not require uniform coverage, we only penalize the overlap between each attention distribution and the coverage so far – preventing repeated attention. Finally, the coverage loss, reweighted by some hyperparameter λ, is added to the primary loss function to yield a new composite loss function:

**Losst = −logP(wt\* ) +λ ∑i min(ait , cti ) (13)**

# **5. IMPEMENTATION**

# **5.1 Dataset**

The dataset used contains over 3,00,000 news articles.

* These news articles belong to two prominent news agencies:

1. CNN
2. Dailymail

The source of the dataset is: [www.tensorflow.org/datasets/catalog/cnn\_dailymail](http://www.tensorflow.org/datasets/catalog/cnn_dailymail)

* The distribution of the dataset will be as follows:

1. Testing: 10,000 articles.
2. Validation: 10,000 articles.
3. Training: Rest of the dataset (Around 2,90,000).

Also, since the number of Dailymail articles(2,19,000) is greater than the number of CNN articles(92,000), the data from both sets have been divided proportionately in the aforementioned distribution (Testing, Validation, and Training).

## **5.1.2 Dataset Preparation**

* Before using the dataset we first prepare the data to be ready for fast and efficient processing.
* We first tokenize the data, then we convert our data into binary format, and then we divide the data into many files for our convenience.
* Since we are using Recurrent Neural Networks, we use sequential data in the form of tokens.
* To generate these tokens we perform tokenization of our articles as well as reference summaries.
* For tokenization, we use Stanford NLP Tokenizer which is a software that has been developed by Stanford University.
* We also count the number of time each token appears in the dataset.
* With this we create a vocabulary file which list the token(word) and its frequency over the dataset.
* Next, we convert the tokenized files into .bin(binary) files.
* This is done to enable better efficiency and speedup the training process.
* We generate three files:

1. Train.bin
2. Valid.bin
3. Test.bin

* The binary files are then divided into smaller files such that each files contains 1000 articles.
* This is done for our convenience so that we can we can use only specific files or less data manually if required.
* Thus following conversions take place:

1. Train.bin -> Train001.bin, Train002.bin, Train003.bin, …
2. Valid.bin -> Valid001.bin, Valid002.bin, Valid003.bin, …
3. Test.bin -> Test001.bin, Test002.bin, Test003.bin, …

# **5.2 Hyper Parameters Values**

* The following are the values of hyperparameters used in our model:
* Batch size= 16
* Word embeddings dimension= 128
* Vocabulary size= 50,000
* Learning rate= 0.25
* Maximum encoder steps= 400
* Maximum decoder steps= 100

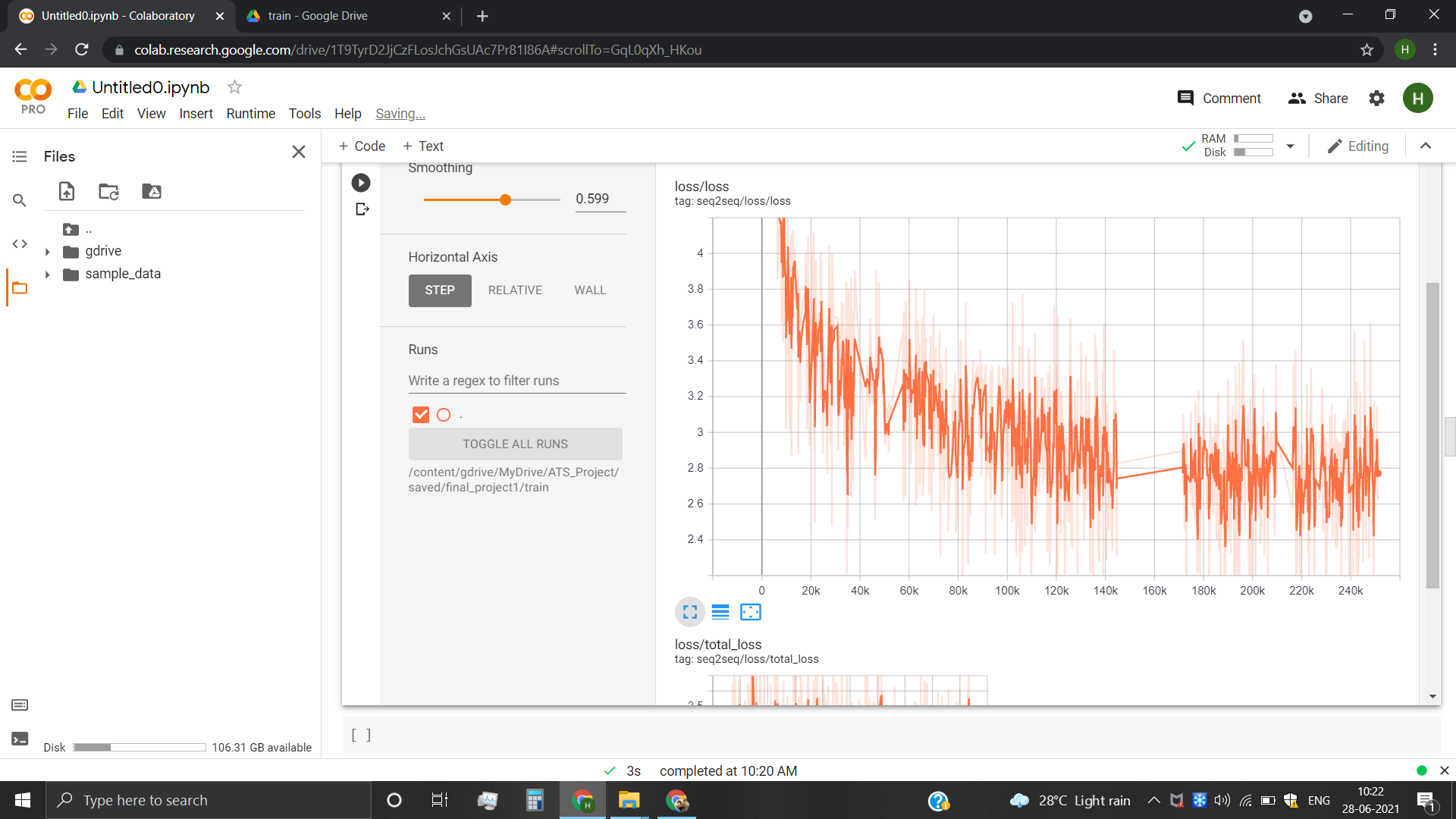
# **5.3 Tensorflow Model**

* Tensorflow is a library in python which enables us to build machine learning models.
* We created a tensorflow model for our summarization training and validation.
* This model is training the dataset to generate weights and thus give us the optimal text summarizer.
* We use the function: tensorflow.nn.bidirectional\_rnn() for our bidirectional LSTM encoder.

# **5.4 Experimental Results:**

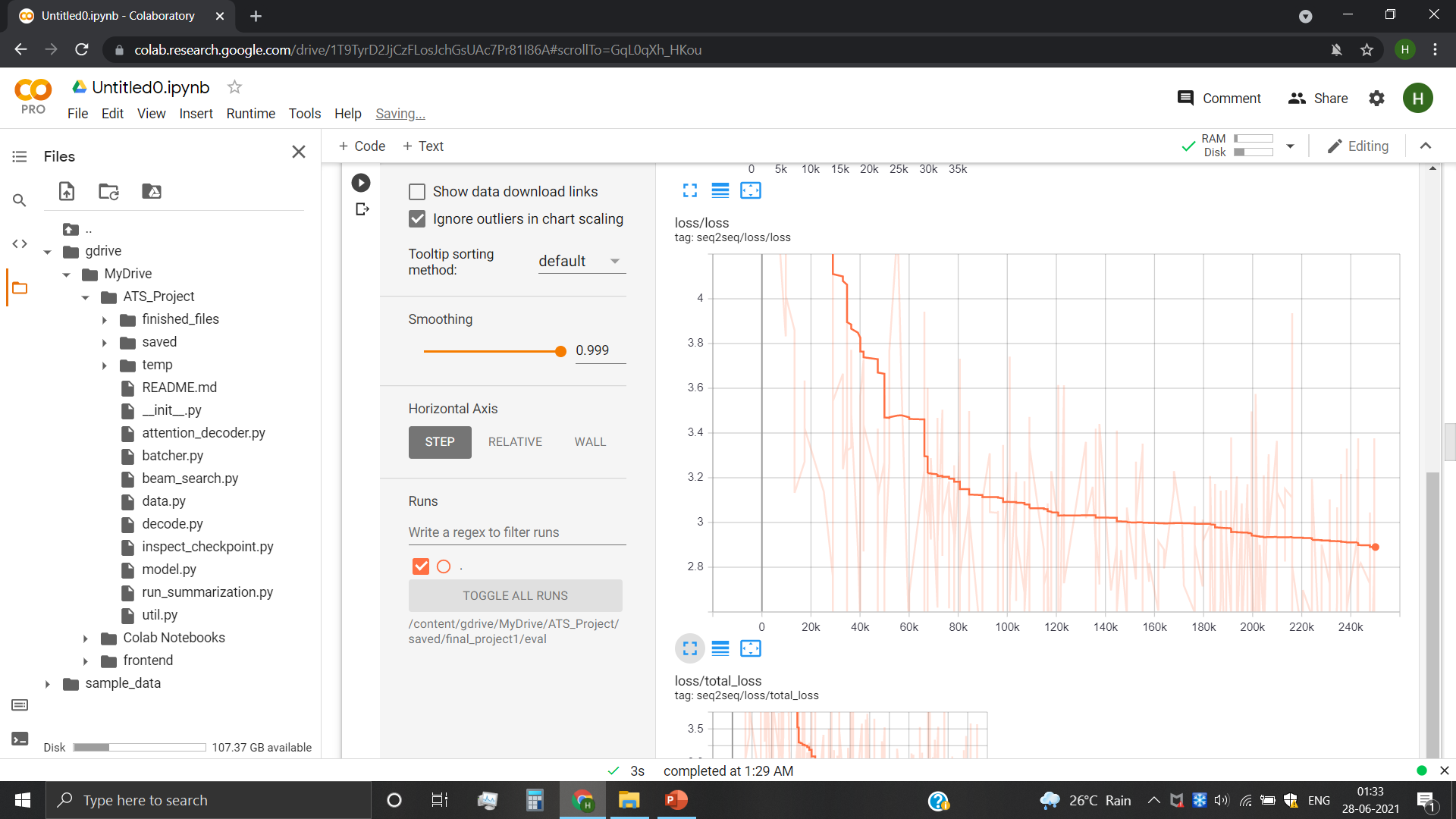
* The model was trained using Google Colaboratory on around 3,00,000 news articles.
* It was trained for around 2,50,000 iterations (around 15 epochs).
* The model was validated on another 10,000 articles to check for overfitting.
* The model was also tested on 10,000 articles.

**Training Loss Graph**



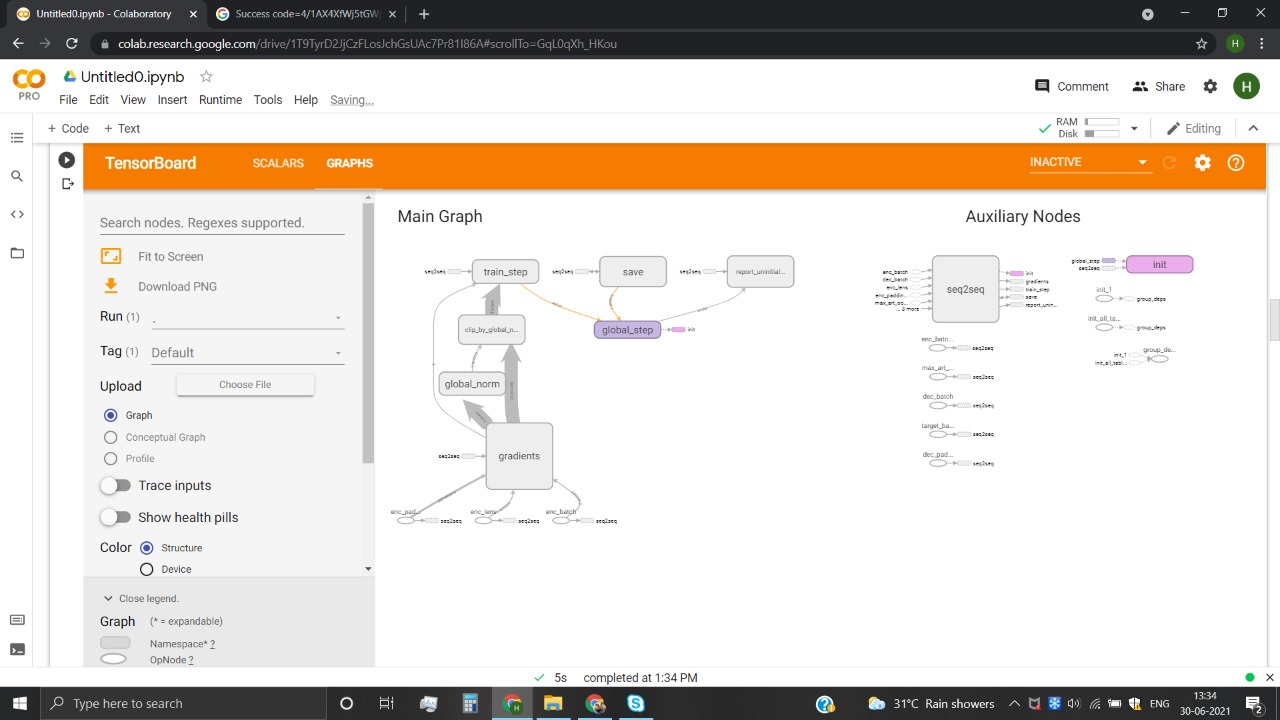
As we can see from the training loss graph, the loss began at a value of around 10 and after 250,000 iterations it went as low as 2.5.

**Validation Loss Graph**

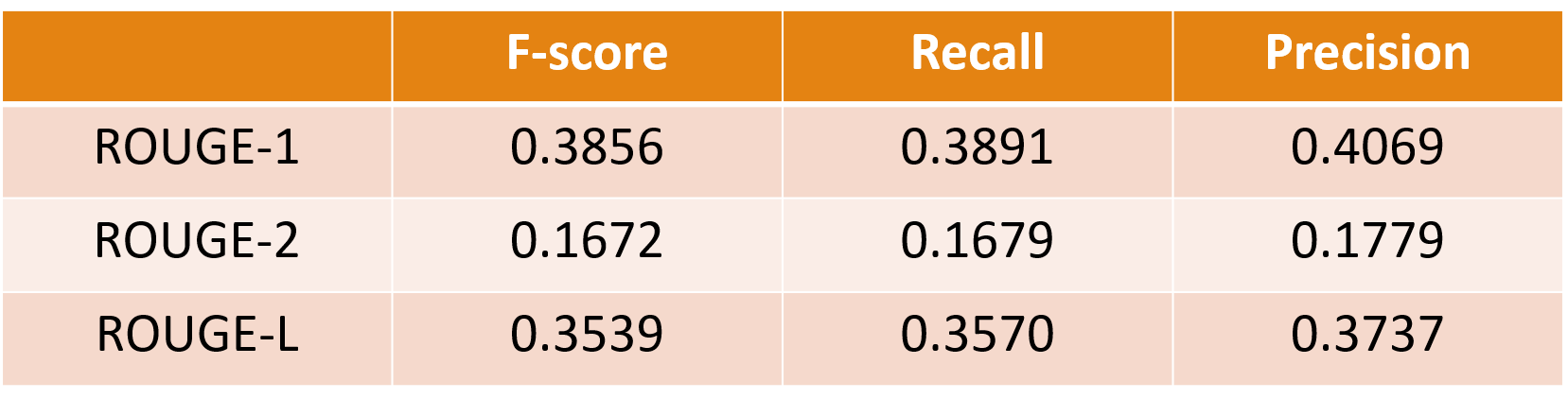


As we can see from the validation loss graph, the graph exhibits similar behavior to training loss graph. Hence, there is no case of overfitting.

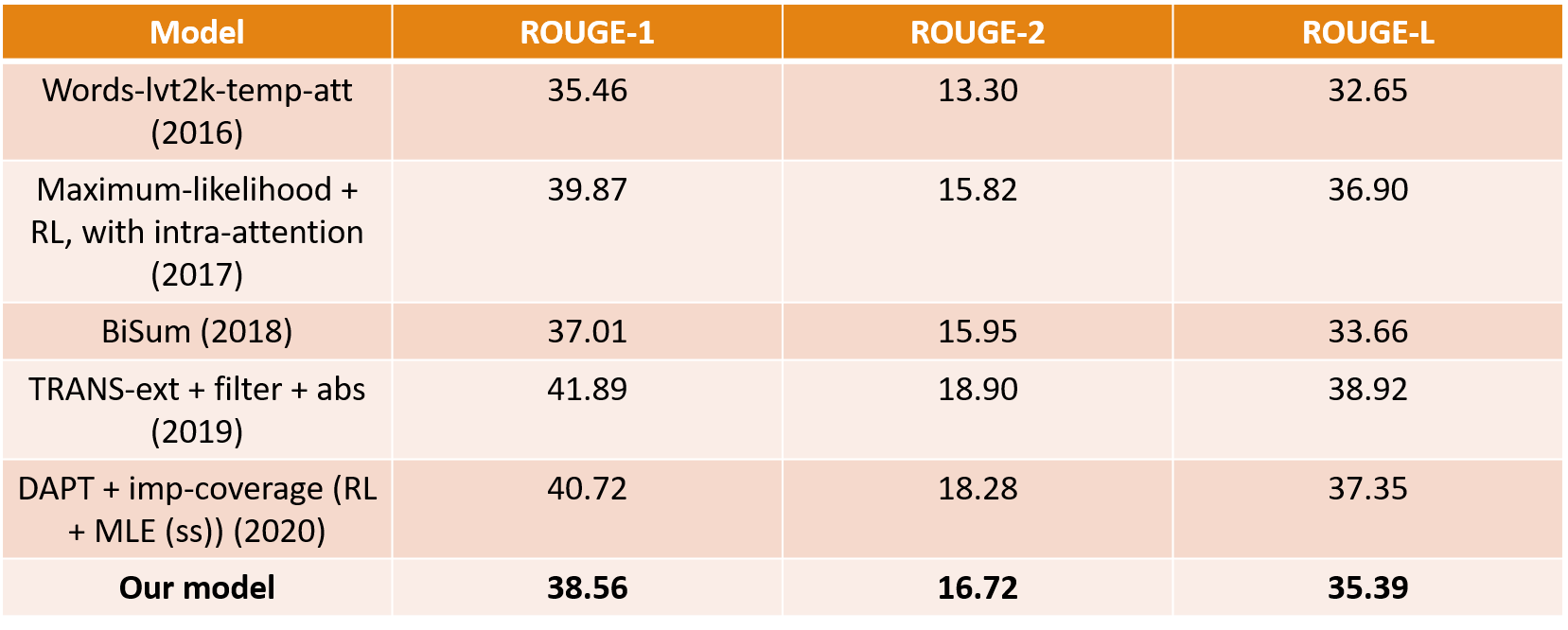
**Visualization of Graph using tensorBoard:-**

****

**ROUGE Results:-**



A brief comparison with some other models :-



**Sample Result :-**

**Article:-**

Former Vice President Walter Mondale was released from the Mayo Clinic on Saturday after being admitted with influenza, hospital spokeswoman Kelley Luckstein said.

"He's doing well. We treated him for flu and cold symptoms and he was released today," she said.

Mondale, 87, was diagnosed after he went to the hospital for a routine checkup following a fever, former President Jimmy Carter said Friday.

"He is in the bed right this moment, but looking forward to come back home," [Carter said](http://nobelpeaceprizeforum.org/live/) during a speech at a Nobel Peace Prize Forum in Minneapolis.

"He said tell everybody he is doing well."

Mondale underwent treatment at the Mayo Clinic in Rochester, Minnesota.

The 42nd vice president served under Carter between 1977 and 1981, and later ran for President, [but lost to Ronald Reagan.](http://www.cnn.com/2013/08/30/us/walter-mondale-fast-facts/) But not before he made history by naming a woman, U.S. Rep. Geraldine A. Ferraro of New York, as his running mate.

Before that, the former lawyer was a U.S. senator from Minnesota.

**Reference Summary :-**

walter mondale was released from the mayo clinic on saturday , hospital spokeswoman said .

the former vice president , 87 , was treated for cold and flu symptoms .

**Generated Summary :-**

the former vice president was released on saturday after being admitted with influenza .

the 87 - year - old was diagnosed with a routine checkup following a fever .

he was also on a nobel peace prize forum in minneapolis .

# **6.1 Conclusion**

* In this project we used a hybrid pointer- generator model to reduce inaccuracies in generated summaries.
* We also used coverage mechanism to avoid repetition of phrases and words.
* Our model exhibits some abstractive abilities, but achieving more abstraction without compromising inaccuracies is yet to be achieved.

# **6.2 Suggestions for future work**

* We first need a more detailed dataset containing multiple reference summaries. This will enable much better ROUGE evaluation and comparison with other models.
* Need to address the issue of difference in vocabulary probabilities and attention probabilities.

# **6.3 References**

1. “Just News It: Abstractive Text Summarization with a Pointer-Generator Transformer” : Vrinda Vasavada, Alexandre Bucquet.
2. “Neural Abstractive Text Summarization with Sequence-to-Sequence Models” : TIAN SHI, YASER KENESHLOO, NAREN RAMAKRISHNAN, CHANDAN K. REDDY.
3. “Get To The Point: Summarization with Pointer-Generator Networks” : Abigail See, Peter J. Liu, Christopher D. Manning 2017 .
4. “Abstractive sentence summarization with attentive recurrent neural networks.” : Sumit Chopra, Michael Auli, Alexander M Rush.
5. “NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE”: Dzmitry Bahdanau, KyungHyun Cho, Yoshua Bengio
6. “Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond”: Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, Bing Xiang
7. Dataset: [www.tensorflow.org/datasets/catalog/cnn\_dailymail](http://www.tensorflow.org/datasets/catalog/cnn_dailymail)
8. Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with at- tentive recurrent neural networks. In *North Amer- ican Chapter of the Association for Computational Linguistics*.
9. Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. 2016. Neural headline generation on abstract meaning representation. In Empirical Methods in Natural Language Processing.
10. Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, C¸ aglar Gulc¸ehre, and Bing Xiang. 2016. Abstrac- tive text summarization using sequence-to-sequence RNNs and beyond. In *Computational Natural Lan- guage Learning*.
11. Philipp Koehn. 2009. *Statistical machine translation*. Cambridge University Press.
12. Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In Neural Information Processing Systems.
13. Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. SummaRuNNer: A recurrent neural network based sequence model for extractive summarization of documents. In *Association for the Advancement of Artificial Intelligence*.
14. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Ben- gio. 2015. Neural machine translation by jointly learning to align and translate. In *International Con- ference on Learning Representations*.
15. Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sen- tence summarization. In *Empirical Methods in Nat- ural Language Processing*.