

# Bert-Text Abstractive Summarization

JOSH COWARD

Boise State University  
joshcoward@u.boisestate.edu

RYAN PACHECO

Boise State University  
ryanpacheco413@u.boisestate.edu

SAJIA ZAFREEN

Boise State University  
sajiazafreen@u.boisestate.edu

April 26, 2021

## Abstract

*Text summarization is a process/technique to shorten an extended text such as books, news articles, blog posts, research papers, emails, and tweets in a coherent, fluent summary. It is one of the Natural Language Processing applications, which significantly impacts our lives with a spiking demand every day. With growing digital media and publishing, text summarization helps with deciding on whether an article/ document/ book is valuable or not. With the availability of textual data, it has become easier to train a model for text summarization. This paper discusses Text Abstractive Summarization using Bert to create a model that produces results comparable to Google's massive Pegasus model they developed and released back in June of 2020. We used the English language for our project as it's a language we can speak and understand. We also used the datasets to train our model consist of only English entries. We have used the CNN/Daily Main, SAMSum Corpus, and BillSum Corpus as our datasets.*

## I. INTRODUCTION

Text Summarization summaries extended text while keeping coherent context. The increasing amount of unstructured data needs to be extracted in relevant summaries. There are different types of text summarization: Extractive and Abstractive. Extractive summarization generates a summary by selecting salient sentences or phrases from the source text, while abstractive methods paraphrase and restructure sentences to compose the summary. We focus on abstractive summarization in our work. There are many abstractive approaches based on a sequence-to-sequence model with a copy mechanism. However, some there are issues with these abstractive methods are: 1) because these methods use a left-context decoder while predicting each word, these methods do not have complete context. 2) as they do not use pre-trained contextualized language models on the decoder, the decoded does learn summary

representation, context interactions, and language modeling together. BERT, which stands for Bidirectional Encoder Representations from Transformation, is designed to pre-train deep bidirectional representations from the unlabeled text by jointly conditioning both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks, including pushing the GLUE score to 80.5% (7.7% point absolute improvement), MultiNLI accuracy to 86.7% (4.6% absolute improvement), SQuAD v1.1 question answering Test F1 to 93.2 (1.5 points absolute improvement) and SQuAD v2.0 Test F1 to 83.1 (5.1 points absolute im-

provement).<sup>1</sup>.

Bert has been very successful in textual entailment, name entity recognition, and machine reading comprehension. We have used the Bert-base-uncased pre-trained model for both the encoder and decoder, with the goal in mind of creating a model that resembles to Google's massive Pegasus model.

## II. BACKGROUND

### i. Works Cited

We have used [1], [2] as reference. In the [1] paper, the text summarization model is based on the Transformer architecture. The architecture adopts the original model of Vaswami et al. (2017). On top of the decoder, they used a Pointer-Generator to increase the extractive capabilities of the network. In Paper [2], proposed 1. A natural language generation model based on BERT, making good use of the pre-trained language model in the encoder and decoder process, and the model can be trained end-to-end without handcrafted features. 2. A design a two-stage decoder process. In this architecture, our model can generate each word of the summary considering both sides' context information. 3. To conduct experiments on the benchmark datasets CNN/Daily Mail and New York Times. Our model achieves a 33.33 average of ROUGE-1, ROUGE-2 and ROUGE-L on the CNN/Daily Mail, which is state-of-the-art. On the New York Times dataset, our model achieves about 5.6% relative improvement over ROUGE-1. Our work doesn't differ much from the papers we referenced here [1], [2]. As BERT can't process sequences with more than 512 tokens, to deal with that our model is using overlapping subsection text of length 512 words and generating summaries for each of the individual subsections. In the end, a final summary across all those summaries. In the case of summaries, we didn't use a pre-trained summary model until the end.

<sup>1</sup>See "Computation and Language", arXiv:1810.04805

### ii. Datasets

Our project used three datasets: **1) CNN/Dailymail**, **2) SAMSum corpus**, and **3) Billum corpus**. There are different datasets for text summarization. We tried to find datasets with "gold" summaries to compare our generated summaries with the "gold" summaries for our approach.

The **CNN/Dailymail** is an English-language dataset containing just over 300k unique news articles written by journalists at CNN and the Daily Mail. We have used version 3.0.0, which can be used to train both abstractive and extractive summarization.

The data consists of news articles and highlight sentences. In the question answering setting of the data, the articles are used as the context and entities are hidden one at a time in the highlight sentences, producing Cloze style questions where the goal of the model is to correctly guess which entity in the context has been hidden in the highlight. In the summarization setting, the highlight sentences are concatenated to form a summary of the article. The CNN articles were written between April 2007 and April 2015. The Daily Mail articles were written between June 2010 and April 2015. The model performance is measured by how high the output summary's ROUGE score for a given article is compared to the highlight written by the original article author. Zhong et al. (2020) report a ROUGE-1 score of 44.41 when testing a model trained for extractive summarization. See the Papers With Code leaderboard for more models.<sup>2</sup>. There is a string for the article for each instance, a string for the highlights, and a string for the id. The train set has 287,113, the validation set has 13,368, and the Test set has 11,490 instances in Split.

The **SAMSum** dataset contains about 16k messenger-like conversations with summaries. Conversations were created and written down by linguists fluent in English. The dataset reflects the proportion of topics of real-life

<sup>2</sup>See "Get To The Point: Summarization with Pointer-Generator Networks", <https://www.aclweb.org/anthology/P17-1099>

messenger conversations. It has 16369 conversations distributed uniformly into 4 groups. Each instance contains an id string, a summary string, and a dialogue string. The train set has 14,732, the validation set has 818, and the test set has 819 instances in Split. The summaries are annotated and assume they should 1) be short, 2) extract important pieces of information, 3) include names of interlocutors, and 4) be written in the third person. Each dialogue has only one reference of summary.<sup>3</sup>

The **BillSum** dataset summarizes the state bills of US Congressional and California states. It has several features: 1)text: bill text, 2) summary: summary of the bills, 3)title: title of the bills, 4)text\_len: number of chars in text, 5)sum\_len: number of chars in summary. The data field has a text string, a [1] summary string, and a title string. The train set has 18,949, the ca\_test set has 1237, and the test set has 3269 instances in Split.<sup>4</sup>

### III. MODEL AND APPROACH

In the encoding and generation process, the attention mechanism is used to concentrate on the most important positions of text. The learning objective of most sequence-to-sequence models is to minimize the negative log likelihood of the generated sequence as shown in following equation, where  $y_t^*$  is the t-th ground-truth summary token.

$$Loss = -\log \sum_{t=1}^{|y|} P(y_t^* | y_{<t}^*, X) \quad (1)$$

We used BERT and tried to fine tune it. BERT consists of several layers. In each layer, there is a first multi-head self-attention sub-layer and then a linear sub layer with residual connection. In each self-attention sub-layer the attention scores  $e_{ij}$  are first calculated as Eq. (2), (3), in which  $d_e$  is output dimension, and  $W^Q$ ,  $W^K$ ,  $W^V$  are parameter matrices.

$$a_{ij} = \frac{(h_i W^Q)(h_j W^K)^T}{\sqrt{d_e}} \quad (2)$$

$$e_{ij} = \frac{\exp e_{ij}}{\sum_{k=1}^N \exp e_{ik}} \quad (3)$$

Then the output is calculated as Eq. (4), which is the weighted sum of previous outputs  $h$  added by previous out-put  $h_i$ . The last layer outputs are context encoding of input sequence.

$$o_i = h_i + \sum_{j=1}^N e_{ij}(h_j W^V) \quad (4)$$

Despite the wide usage and huge success, there is also a mismatch problem between these pre-trained context encoders and sequence-to-sequence models. The issue is that while using a pre-trained context encoder like BERT, they model token-level representations by conditioning on both direction context. During pre-training, they are fed with complete sequences. However, with a left-context-only decoder, these pre-trained language models will suffer from incomplete and inconsistent context and thus cannot generate good enough context-aware word representations, especially during the inference process.

Based on the sequence-to-sequence framework built on top of BERT, we first followed the design mentioned in [2] which is a refine decoder at word-level to tackle the two problems described in the above section. We also introduce a discrete objective for the refine decoders to reduce the exposure bias problem.

<sup>3</sup>See "SAMSum Corpus: A Human-annotated Dialogue Dataset for Abstractive Summarization", <https://www.aclweb.org/anthology/D19-5409>

<sup>4</sup>See "BillSum: A Corpus for Automatic Summarization of US Legislation", 1910.00523

#### IV. EVALUATION

#### V. IMPLICATIONS

#### VI. RESULTS

#### VII. CONCLUSION

#### REFERENCES

- [1] Dmitrii Aksenov et al. “Abstractive Text Summarization based on Language Model Conditioning and Locality Modeling”. In: *arXiv preprint arXiv:2003.13027* (2020).
- [2] Haoyu Zhang, Jianjun Xu, and Ji Wang. “Pretraining-based natural language generation for text summarization”. In: *arXiv preprint arXiv:1902.09243* (2019).