CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

Abstractive Text Summarization using NLP

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science
In Computer Science

By
Poojan Patel

May 2023
The thesis of Poojan Patel is approved:

Prof. Robert McIlhenny, Ph.D

Prof. Mahdi Ebrahimi

Dr. Taehyung Wang, Ph.D Chair

California State University, Northridge
Acknowledgment

I would like to express my sincere gratitude to all those who have contributed to the completion of my thesis.

First and foremost, I would like to thank my thesis advisor, Dr. Taehyung Wang, for his invaluable guidance, insightful comments, and unwavering support throughout the entire research process. Without his expertise and encouragement, this thesis would not have been possible.

I am also grateful to the members of my thesis committee, Dr. Mahdi Ebrahim and Dr. Robert McIlhenny, for their constructive feedback and helpful suggestions on improving the quality of my research.

I would like to thank my family for their unwavering support, encouragement, and understanding throughout my academic journey. Their love and constant encouragement have been a source of motivation for me.
# Table of Contents

Signature Page ........................................... ii

List of Figures ....................................... vi

List of Acronyms ..................................... vii

Abstract ............................................. viii

1. Introduction ........................................ 1
   1.1 Problem Statement ............................... 1
   1.2 Objective ....................................... 2
   1.3 Background ..................................... 3
   1.4 Target .......................................... 4

2. Abstractive Text Summarization Documentation ......... 5
   2.1 Introduction ..................................... 5
   2.2 Recent Works ................................... 6

3. Models And Results ................................ 10
   3.1 Transformer Model .............................. 10
   3.2 BERT Model .................................... 14
   3.3 PEGASUS Model ................................. 17
   3.4 Data Sets ....................................... 19
   3.5 Hardware Requirements ........................ 20

4. Implementations .................................... 22

5. Results and Evaluation ............................. 30
   5.1 Results ......................................... 30
       5.1.1 Transformer Output ......................... 30
       5.1.2 BERT Output ................................ 31
       5.1.3 PEGASUS Output ........................... 32
5.1.4 Comparison on Sample Texts 34
5.2 Evaluation 36
  5.2.1 ROUGE Metrics 37
5.3 Limitations Of Models 41
  5.3.1 Transformer Model 41
  5.3.2 BERT Model 41
  5.3.3 PEGASUS Model 42
6. Future Work 43
7. Conclusion 44
8. References 45
## List of Figures

| Figure (i) | Multi-head Attention | 11 |
| Figure (ii) | Transformer model Architecture | 12 |
| Figure (iii) | BERT Model Architecture | 16 |
| Figure (iv) | PEGASUS model Architecture | 19 |
| Figure (v) | Inshorts dataset after preprocessing | 24 |
| Figure (vi) | Transformer model validation graph on 15 epochs | 25 |
| Figure (vii) | Transformer model validation graph on 178 epochs | 26 |
| Figure (viii) | Error while generating the model | 27 |
| Figure (ix) | Successful training model of BERT Model on 3 Epochs | 37 |
| Figure (x) | BERT model training failure due to less GPU power | 28 |
| Figure (xi) | Successful PEGASUS model | 29 |
| Figure (xii) | Transformer model output | 30 |
| Figure (xiii) | Output of Transformer Model with 277 epochs | 31 |
| Figure (xiv) | BERT model outputs | 32 |
| Figure (xv) | PEGASUS model Output | 34 |
| Figure (xvi) | Code snippet for evaluating model | 39 |
## List of Acronym

1. **NLP**  
   Natural Language Processing
2. **DL**  
   Deep Learning
3. **RNN**  
   Recurrent Neural Network
4. **CNN**  
   Convolutional Neural Network
5. **ROUGE**  
   Recall-Oriented Understudy for Gisting Evaluation
6. **BLEU**  
   Bilingual Evaluation Understudy
7. **BERT**  
   Bidirectional Encoder Representation from Transformers
8. **GloVe**  
   Global Vectors for Word Representation
9. **GPU**  
   Graphical Processing Unit
10. **TPU**  
    TensorFlow Processing Unit
11. **MLM**  
    Masked Language Model
12. **PEGASUS**  
    Pre-training with Extracted Gap-sentences for Abstractive Summarization
Abstract

Abstractive Text Summarization Using NLP

By Poojan Patel

Master of Science in Computer Science

This paper aims to compare three natural language processing models for abstractive text summarization: the Transformer, BERT, and PEGASUS models. Abstractive summarization is the process of generating a condensed and coherent summary of a longer document, which captures the most important information and ideas of the original text.

The paper provides a detailed description of the three models, including their architecture, training objectives, and pre-training methods. A type of model known as the Transformer utilizes attention mechanisms to produce summaries and operates on sequences in a sequence-to-sequence fashion. BERT is an auto-encoder model that utilizes both forward and backward information flow through a bidirectional encoder, and a left-to-right decoder to produce summaries. PEGASUS is a pre-training method that uses a masked language modeling objective to generate high-quality summaries.

The paper compares the three models based on several evaluation metrics, such as ROUGE, BLEU scores. The evaluation is conducted on multiple datasets, including CNN/DailyMail, InShorts. In terms of ROUGE scores, the outcomes indicate that the three models surpass prior state-of-the-art models. Across all datasets and evaluation metrics, PEGASUS consistently achieves the highest scores.

The paper also analyzes the strengths and weaknesses of each model. The Transformer model is computationally efficient and can generate summaries quickly, but it sometimes generates
incomplete or redundant summaries. BERT can generate fluent and coherent summaries, but it struggles with generating summaries that capture the most important information. Although computationally expensive and demanding a substantial amount of pre-training data, PEGASUS produces informative and coherent summaries of high quality. To summarize, this article presents a thorough evaluation of three advanced models for abstractive text summarization - Transformer, BERT, and PEGASUS. While each model exhibits impressive results, PEGASUS performs the best overall. Nonetheless, determining the optimal model is contingent upon the specific use case and the computational resources available.
1. Introduction

1.1 Problem Statement

As the amount of data and published material on the internet grows, the need to swiftly acquire and comprehend relevant information has become an ongoing research area. The work of summarizing data is difficult, and the large amount of information available on the internet can be intimidating. Text summarization, which reduces a document's length while maintaining its meaning and content, became popular. In light of this, a summary is valuable since it may retrieve large volumes of document material while also saving time. It was carried out manually before, but automation has brought about a lot of benefits recently.

Creating a succinct and cohesive summary of a larger article while retaining the most crucial information is a difficult job for natural language processing (NLP) systems. Deep learning models including encoder-decoder architectures [7] and transformer models [1], as well as methods like attention mechanisms (Bahdanau et al., 2015) and reinforcement learning, have all been studied in previous research (Paulus et al., 2018). Despite the positive outcomes, these techniques still have trouble producing summaries that are accurate and coherent, especially when faced with ambiguous or complex information (Wang et al., 2019). The challenge for abstractive text summarization with NLP is to create more reliable and effective models that can produce high-quality summaries that are true to the original text while yet being brief and simple to read.
1.2 Objective

The objective of this dissertation is to evaluate and contrast the effectiveness of transformer-based models, specifically BERT and PEGASUS, in the field of natural language processing, concerning abstractive text summarization. ATS seeks to provide a condensed version of a text while retaining the key points. In NLP applications, such as abstractive text summarization, transformer-based models have demonstrated promising outcomes. PEGASUS is a transformer model that has been pre-trained with an encoder-decoder architecture and a self-attention mechanism. On the other hand, BERT is a denoising autoencoder that can be tailored for various NLP tasks.

PEGASUS and BERT will be tested against conventional encoder-decoder transformer models and other cutting-edge abstractive summarization models using a dataset of news items and other datasets for the study. The effectiveness of the generated summaries will be assessed using a variety of evaluation measures, ROUGE (Recall-Oriented Understudy for Gisting Evaluation)[32]. Also, the study will examine the benefits and drawbacks of the PEGASUS and BERT models, pinpoint the variables that influence how well they function, and suggest potential improvements and alterations that could be made to boost their summarization abilities.

This thesis's overarching objective is to advance transformer-based NLP models for abstract text summarization. The purpose of the study is to offer insights on the efficacy of transformer-based models for abstractive text summarization and to suggest changes that could be made to these models. As a result, this investigation may contribute to the advancement of more accurate and effective summarization technologies, which can be
beneficial in various areas, such as news summarization, document summarization, and chatbot responses.

1.3 Background

In the field of natural language processing, abstractive text summarization plays a crucial role by producing a condensed version of a given text while preserving the essential information. Abstract techniques can offer a better understanding of language comprehension because the algorithm must ensure coherence and grammatical accuracy of the generated sentences during the training phase [6]. The approaches used in abstractive summarization have mainly relied on heuristics based on linguistic and syntactic knowledge of language [7].

Recently, the architecture of encoder-decoder models has demonstrated its effectiveness in both interpreting and generating sequential data, particularly text. One component of this architecture is the encoder, which is responsible for reading and encoding the input into a latent vector. The decoder, on the other hand, reads and interprets the data represented by this vector, generating a new sequence of text. [8]. One approach has been to combine encoder-decoder models with an attention mechanism for tasks involving language generation, such as machine translation and abstractive text summarization [9-12]. Sequence to sequence (seq2seq) models, which blend attention with the encoder-decoder architecture, have sparked more interest in abstractive summarization [13-15]. The Transformer, developed by Vaswani et al. [1], reengineered the RNN-based encoder-decoder model, relying more on the attention mechanism. Transformer is an unique approach that makes use of a combination of attention mechanisms and feedforward
layers, eliminating the need for an RNN component. This technique outperforms prior techniques in terms of decreasing training time and improving translation quality in machine translation jobs, especially on long text sequences. Because the two difficulties are so similar, the Transformer has the potential to excel at abstractive summarization[13].

To sum up, abstractive text summarization is a crucial natural language processing task that entails producing a summary of a given text while preserving its essential information. The utilization of attention and the encoder-decoder architecture has been critical in driving the growing interest in abstractive summarization, and the Transformer model exhibits promise in attaining top-tier performance in this task.

1.4 Target

The primary goal of this thesis is to assess the performance of three advanced models for abstractive text summarization - BERT, PEGASUS, and Transformers. The research will evaluate the efficacy of these models in producing high-quality summaries while preserving vital information, ensuring grammatical accuracy, and creating coherent and readable summaries. Furthermore, the study will investigate the influence of various training data and parameters on the models' performance. The findings of this research may offer valuable insights into the models' strengths and limitations, aiding in the selection of the most appropriate model for different summarization tasks.
2. Abstractive Text Summarization Documentation

In the Documentation chapter, the explanation of abstractive text summarization is provided. The chapter introduces the reader to prior research in this field, elucidates the essential theoretical concepts, and provides a detailed account of the models used in this research.

2.1 Introduction

Text summarization involves the condensation of a corpus of text into a concise and meaningful set of words that encapsulate the most significant information from the original content. Any textual material, including news and reviews, can be the source. A corpus's compact size allows for quicker comprehension of its contents and makes it simpler to read. So, one benefit of text summary is time effectiveness. Yet, the timing isn't always relevant. While summarizing a text, one tries to retain the main ideas while eliminating as many extraneous details as possible. The result is that a summary is more straightforward and clearer. Another justification for using a summary is that it eliminates misunderstanding and confusion. As previously mentioned, automatic text summarization can be accomplished in two ways: extractive and abstractive. The technique used to extract information from the document and construct the summary distinguishes them.

There are various types of summaries available, as listed below:

1. **Extractive Summary** - Extractive summarization aims to extract significant sentences or words from text to present essential information. This approach is straightforward and produces summaries that are grammatically and syntactically accurate.

2. **Abstractive Summary** - Abstractive summarization methods are used to generate more flexible and consistent summaries. In this
3. Method tactics include paraphrase, inclusion of real-world information, and sentence coherence. The phrases in this output are created by the model rather than taken from the original text data. Abstractive text summarization more closely resembles the thought process of humans. The individual reads the text, matches it to their knowledge and related information, and then condenses its main points into a concise summary. As a result, generating a summary through abstractive methods is more difficult than using an extractive approach because the model must deconstruct the source text into individual tokens and reconstruct them into coherent sentences. Developing accurate and grammatically correct summaries requires highly precise and sophisticated models.

2.2 Recent Works
Transformer model for summarizing was first used by Liu et al. (2018). To increase the input sequence size, only the decoder was utilized in addition to the extraction model. Zhang et al. (2019) enhanced the Transformer-based model by incorporating BERT. Utilizing the mask learning approach, they employ a two-stage process. Some people try to add an extractive model to their abstractive summarization models to make them better. In the area of abstract summarization, encoder-decoder architectures founded on RNNs (Chung et al., 2014; ) and more recently, Transformers have become the dominant frameworks [6]. According to [45], prior research on neural abstractive summarization achieved success by utilizing substantial, high-quality datasets that contained guided document-summary pairs [45][1]. Over the last few years, there has been an increasing interest in building novel datasets for summarization, which include abstractive summaries[28], lengthy documents (Cohan et al., 2018), multiple documents [47], and varied topics [47][49]. In this paper, we focus solely on pre-training
targets for abstractive text summarization and evaluate them on 12 downstream datasets that include news, science, short stories, instructions, emails, patents, and legislative legislation [46][26][46][12][45]. As a pre-training target for downstream summarization tasks, a self-supervised objective called Gap Sentences Generation (GSG) was developed, in which full sentences were concealed from the document and the missing sentences were formed from the rest of the document. They discovered that choosing salient sentences outperformed choosing random or lead sentences. This strategy is similar to the downstream goal in that it encourages summary-like generation and whole-document understanding, making it perfect for abstractive summarization. PEGASUS method, Pre-training with Extracted Gap-sentences for Abstractive Summarization Sequence-to-Sequence models, or PEGASUS, pre-trains a Transformer encoder-decoder using a huge corpus of documents (Web and news articles). Rush, Chopra, and Weston[12] devised an encoder-decoder model incorporating attention, which was one of the first approaches to implementing neural methods for end-to-end abstractive summarization; this concept is based on previous work in machine translation [11].
<table>
<thead>
<tr>
<th>Year</th>
<th>Authors</th>
<th>Techniques</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2016</td>
<td>Ramesh Nallapati, Cicero Nogueira dos Santos, Bowen Zhou, Caglar Gulcehre and Bing Xiang [28]</td>
<td>A decoder-encoder model with high attention and a large vocabulary.[28]</td>
<td>Employed the framework of Attentional Encoder-Decoder Recurrent Neural Network outperforming the model on two different corpora.[28]</td>
</tr>
<tr>
<td>2017</td>
<td>Abigail See, Christopher D. Manning and Peter J. Liu and [31]</td>
<td>Abstractive Text Summarizer using pointer generator network and coverage mechanism.[31]</td>
<td>Used a pointer-generator network that would copy words from the source text through pointing and Coverage was used to keep track of data which had been summarized. [31]</td>
</tr>
<tr>
<td>2017</td>
<td>Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin [1]</td>
<td>Encoder-decoder model which only depends on attention mechanism[1]</td>
<td>Sequence transduction models use recurrent or convolutional neural networks to connect the encoder and decoder through attention and the transformer, based only on attention mechanisms, dispensing with recurrence and convolution entirely.[1]</td>
</tr>
<tr>
<td>2019</td>
<td>Yang Liu and Mirella Lapata [6]</td>
<td>Text Summarization with the pretrained model (BERT). [6]</td>
<td>For abstractive summarization, different optimizers were used for the encoder and the decoder to alleviate the mismatch between the two. [6]</td>
</tr>
<tr>
<td>Year</td>
<td>Authors</td>
<td>Method</td>
<td>Results</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>--------</td>
<td>---------</td>
</tr>
<tr>
<td>2019</td>
<td>Jingqing Zhang, Yao Zhao, Mohammad Saleh, Peter J. Liu [7]</td>
<td>Pre-training with extracted gap-sentences for abstractive text summarization[7]</td>
<td>Examining the best PEGASUS model on 12 summarizing tasks from various domains yields cutting-edge results on all datasets, including low-resource summarization.[7]</td>
</tr>
<tr>
<td>2020</td>
<td>Joshua Maynez, Shashi Narayan, Bernd Bohnet, Ryan T. McDonald[35]</td>
<td>A large scale human evaluation of several neural abstractive summarization systems to better understand the types of hallucinations they produce.[35]</td>
<td>The paper analyzes the issues with these models when applied to abstractive document summarization, which tend to generate hallucinated content. However, pretrained models are better at generating factual and faithful summaries as confirmed by humans and textual entailment measures, leading to potential advancements in evaluation metrics and training for neural text generation models.</td>
</tr>
</tbody>
</table>
3. Models and Definitions

This section describes the description of the various models from the beginning, including problems during development and how they were resolved and definitions.

3.1 Transformer model

The multi-layer encoder-decoder architecture of the Transformer model includes several attention layers, fully connected feed-forward networks, normalization layers, and residual connections, stacked on top of each other. Two types of attention mechanisms are employed, with dot products used to calculate attention scores. The initial mechanism is utilized only in the decoder and evaluates attention scores between queries obtained from the hidden state of the preceding layer, and keys and values extracted from the encoder's output. The second mechanism, referred to as self-attention, computes keys, values, and queries from the hidden states of the previous layer in the Transformer. Multiple heads are applied to the attention mechanism in parallel, allowing for the learning of different contextual representations at each head. The multi-head attention is illustrated in the figure below.
After achieving a new level of excellence in machine translation duties, many advanced models for various natural language processing tasks adopted the Transformer architecture. Nevertheless, although it frequently outperforms RNN- and CNN-based models, the Transformer cannot be deemed superior in every circumstance since each model captures distinct features of the data. The diagram below depicts the entire Transformer architecture.
The Transformer design's key breakthrough is the incorporation of self-attention, enabling the model to focus on distinct components of the input sequence at varying intervals. This is achieved by implementing several attention heads, each of which develops a unique attention mechanism. As a result, the Transformer model can process complex input sequences while maintaining accuracy and efficiency.

Here is the mathematical formulation for the multi-head attention mechanism in the Transformer:
Let $X = x_1, x_2, ..., x_n$ be the input sequence, where each $x_i$ is a $d$-dimensional vector. Let $Q$, $K$, and $V$ be linear projections of $X$, where $Q$, $K$, and $V$ are matrices with dimensions $dxh$, $dxh$, and $dxh'$, respectively\[1\]. Here, $h$ is the number of attention heads, and $h'$ is the dimensionality of the output vectors.

The attention score for the $i$-th head is calculated as follows:

$$\text{Attention}(Q, K, V)_i = \text{softmax} \left( \frac{QW_i(KW_i)^T}{\sqrt{d}} \right) VW_i$$

where $WQ_i$, $WK_i$, and $WVi$ are learnable weight matrices.\[1\]

The outputs of the multiple attention heads are concatenated and passed through a linear layer to produce the final output:

$$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1, \text{head}_2, ..., \text{head}_h)WO$$

where $WO$ is a learnable weight matrix.\[1\]

The Encoder and Decoder both use multi-head attention and feedforward layers. The Encoder also includes positional encoding to incorporate position information into the input embeddings.

The mathematical formulation for the Encoder is as follows:

Let $X = \{x_1, x_2, ..., x_n\}$ be the input sequence, where each $x_i$ is a $d$-dimensional vector.

Let $PE$ be the positional encoding matrix with the same dimensions as $X$.\[1\]

The Encoder consists of a stack of $N$ identical layers. The output of the $i$-th layer is denoted as $H^i = \{h^1, h^2, ..., h^n\}$, where each $h^i$ is a $d$-dimensional vector.

The $i$-th layer of the Encoder consists of the following two sublayers\[1\]:

\[1\] Reference to a specific source or paper is implied but not explicitly stated in the text.
1. Multi-Head Self-Attention:

$$\text{MultiHead}(H^{(i-1)}, H^{(i-1)}, H^{(i-1)}) + H^{(i-1)}$$

2. Feedforward:

$$\text{FFN}(H^{(i-1)}) + H^{(i-1)}$$

where FFN is a feedforward neural network with a ReLU activation function.

The Decoder is similar to the Encoder, except it has an additional sublayer for dealing with the Encoder output. The mathematical formula for the Decoder is as follows[1]:

Let $Y = \{y_1, y_2, ..., y_m\}$ be the output sequence, where each $y_i$ is a $d$-dimensional vector.

The Decoder also consists of a stack of $N$ identical layers. The output of the $i$-th layer is denoted as $H^\sim(i) = \{h_1^\sim(i), h_2^\sim(i), ..., h_m^\sim(i)\}$, where each $h_i^\sim(i)$ is a $d$-dimensional vector. The $i$-th layer of the Decoder consists.

### 3.2 - BERT (Bidirectional Encoder Representations from Transformers)

BERT introduced a significant advancement in technology by utilizing the bidirectional training of Transformer, an attention model widely used in natural language processing. Unlike previous attempts that examined text sequences only from left to right or by combining left-to-right and right-to-left training, BERT's bidirectional approach permits a language model to acquire a more in-depth understanding of language context and flow[46]. The research paper also describes a new technique called Masked LM(MLM), which enables bidirectional training in models where it was not previously feasible[46].

An encoder and a decoder, which both serve a distinct purpose, make up the Transformer. Only the encoder component is required for BERT to achieve its purpose of
building a language model. The accompanying diagram goes into great detail on the Transformer's encoder[46]. A group of tokens are used as the input and are converted into vectors before being fed into the neural network. An array of H-dimensional vectors with each vector representing an input token with the same index is the result. For a range of natural language processing applications, this method produces a model that is both very efficient and effective[46].

To prepare word sequences for BERT, a [MASK] token is inserted to replace 15% of each sequence. BERT then utilizes the surrounding non-masked words in the sequence to infer the original values of the masked words. This process, known as output word prediction, involves using contextual information to make predictions.

1. An additional layer known as the classification layer is incorporated onto the encoder output.

2. The embedding matrix is used to convert the output vectors into the vocabulary dimension by means of multiplication.

3. To calculate the probability of each word in the vocabulary, the softmax function is used[46]. This involves taking the exponential of each score and then dividing it by the sum of exponentials for all the scores. By doing so, we can assign a probability distribution over the vocabulary, with each word having a probability between 0 and 1 [46].
Figure (iii) BERT model architecture

Line pairs are given to the model as input during the training phase[46], and it is taught to predict whether the next sentence in the pair will be the sentence after it in the original document. A sentence from the corpus is randomly chosen as the second sentence in the remaining pairs of inputs, which make up about half of the training inputs. In the other half, the second sentence is the following sentence in the original text. To guarantee that it is unrelated to the first sentence in the pair, a random statement was chosen.

There are difficulties in developing a bidirectional LM based on transformer concepts. The input token cannot perceive itself, according to the multi-layered transformer paradigm. Otherwise, since the target word is a component of the input, predictions are pointless. Davin [23] proposed the Masked Language Model (MLM), which is based on the cloze procedure concept [23]. MLM conceals some words (targets) and uses the
remaining ones to anticipate what will come next rather than using the words that came before. Input tokens designated as targets make up 15% of the total. The rules for replacements are as follows if the i-th token is picked.

- 80% replace it with the token [MASK]: I will become a phd student -> I will become a phd [MASK]
- 10% replace it with a random word: I will become a phd student -> I will become a phd cat
- 10% keep it unchanged: I will become a phd student -> I will become a phd student.

BERT is without any question a breakthrough in machine learning for NLP. The main merit of it is that fine-tuning will likely allow a large number of practical applications in the future.

### 3.3 -PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization

When refined for downstream NLP tasks like text summarization, recent work pre-training Transformers with self-supervised objectives on big text corpora has demonstrated remarkable results. Pre-training goals specific to abstractive text summarization have not, however, been investigated [26]. Additionally, systematic evaluation across several domains is lacking. In this study, we offer a new self-supervised aim for pre-training huge Transformer-based encoder-decoder models on enormous text corpora. Using a method analogous to an extractive summary, PEGASUS creates a single output sequence from the remaining phrases after crucial statements from the source document are hidden or
eliminated. Our top-performing PEGASUS model was tested by researchers on 12 different summarization tasks, including news and science articles, stories, instructions, communications, patents, and statutory bills. The efficiency of PEGASUS in several fields was usefully shown by this evaluation. Tests show that it performs at the cutting edge on all 12 downstream datasets as determined by ROUGE scores [26]. In six datasets with only 1000 samples, this model outperforms past state-of-the-art summarization results, which is another unexpected performance. Finally, to validate their findings and show that this model summaries work as well as people across a range of datasets, the researchers employed human evaluation[26].

Several whole sentences are removed from texts during model training, and the model is then tasked with reintroducing them. A text with missing sentences is provided as the input for pre-training, and the output is created by connecting the missing sentences. This is a difficult assignment, even for people, therefore we don't expect the model to complete it perfectly.

Yet, the goal of this exercise is to teach the model how to extract information from a text in order to create an output that is comparable to the work of fine-tuning summarization.

Overall, the training benefits the model's ability to summarize. The authors acknowledge that, even for people, this endeavor appears to be nearly difficult. Yet, such training results in the creation of phrases that incorporate a passage from the main text, supporting their theory. Gap Sentence Generation, often known as GSG, is this procedure[26].

Also, the authors claim that the ideal method for masking is to pick the most significant sentences from the document. This is accomplished by identifying sentences that, using a metric called ROUGE, are most similar to the entire document (which is usually used to evaluate the quality of a summary in summarization tasks).
3.4 PEGASUS model Architecture

In conclusion, the Pegasus model is a sequence-to-sequence model with gap-sentence generation as a pre-training objective tailored for abstractive text summarization. This model is able to adapt to unseen summarization datasets very quickly, achieving strong results in as little as 1000 examples. This model has successfully achieved summaries which satisfy human performance on multiple datasets using human evaluation.

3.4 Data Sets

A set of text-summary pairs created by humans can be used to train an abstractive neural network-based summarizing system. In order to accomplish this, Nallapati et al. [27] created a dataset from CNN and Dailymail articles that included over 300,000 pairs of human-written summaries. The CNN/Daily Mail dataset is a collection of news stories with multi-sentence summaries taken from the CNN.com and dailymail.com websites.

The justification for using this strategy is that it has come to be the standard for training summarization models, allowing us to compare the results of our work to those of earlier studies. In addition, a crucial area in which we will apply our summarizing
Other datasets used in the Pegasus model are mentioned below:

The XSum dataset[26] comprises 227,000 articles published by BBC between 2010 and 2017. The articles cover diverse topics and are accompanied by well-written one-sentence summaries. The CNN/DailyMail dataset[20] includes 93,000 articles from CNN and 220,000 articles from Daily Mail. Both publishers provide bullet point summaries for their articles. In this study, we use the non-anonymized version of the CNN/DailyMail dataset, which was also utilized in See et al.'s (2017) research.

Inshorts dataset 55.1k rows and 5 columns[19], Inshorts is a news service that gives brief summaries of breaking news from throughout the internet. This dataset includes headlines and summaries of news items, as well as their sources.

### 3.5 Hardware Requirements

Training a large-scale Transformer model like GPT-3 or T5 on a huge corpus like CNN/DailyMail, in general, necessitates significant computing resources, such as a powerful GPU or TPU cluster and a large quantity of memory. For example, the T5 model's creators employed 32 TPUs for pre-training and fine-tuning, each with 8 cores, 64GB of RAM, and a TPU-to-TPU communication bandwidth of 56 Gbps. The largest T5 model's pre-training took 3.14 exaFLOPs of compute, which is equivalent to executing the training for more than 355 years on a single CPU core(Joshi et al., 2020).

The BERT-Large model was pre-trained using 64 TPUv3 cores by the original creators of the BERT model. Each TPUv3 core contains 8 NVIDIA V100 GPUs, 64GB of high-bandwidth memory, and 1.2 TB/s of total memory bandwidth. The pre-training
procedure took about four days to complete[23].

The original PEGASUS model was pre-trained on a TPUv3 Pod with 2048 TPU cores, totaling 16,384 NVIDIA V3 TPUs. Using this hardware combination, the pre-training procedure took 10 days to complete (Zhang et al., 2020). The authors employed a smaller TPUv3 Pod with 32 TPU cores to fine-tune the CNN/DailyMail dataset. Each TPU core contains 8 NVIDIA V3 TPUs, 64GB of high-bandwidth memory, and 1.2 TB/s of total memory bandwidth.

For experimentation of my model training and evaluation has been done using Google cloud Collaboratory with below specifications:

CPU: Colab provides a single virtual CPU core with 2 threads. The CPU is an Intel(R) Xeon(R) CPU @ 2.30GHz.

GPU: Colab provides access to a GPU for certain types of computation. The GPU is an NVIDIA Tesla K80 with 12GB of VRAM.

RAM: Colab provides 12GB of RAM.

Disk space: Colab provides approximately 35GB of disk space in the user's Google Drive account.
4. Implementation

Based on the requirements and the model description I implemented my first transformer model with smaller epochs to compare the already existing model with its outputs. Used below mentioned programming environment and python libraries:

Programming Environment:

- Ubuntu 16.04
- Python 3.6
- re (regular expression libraries) 3.11
- os (Miscellaneous operating system interfaces) 3.2
- numpy 1.2.4
- pandas 1.5.3
- tensorflow 2.12
- keras 2.12.0
- matplotlib 3.7.1
- rogue 1.0.1
- glove

The majority of experiments employed the following hyperparameters, which were primarily derived from OpenNLP documentation recommendations for the summarization task [30]. To accelerate the training process and promote model convergence, the depth, hidden state size, and training step count were decreased. In contrast to [29], authors train a new copy attention layer from scratch rather than reusing an attention layer from the Decoder. I used the Adam optimizer with the Noam decay approach for training [34].
Batch size = 4096 tokens

Validation batch size = 0.1 % of all sample

Hidden state size = 256

Word vector size = 256

Number of layers in Encoder = 4

Number of layers in Decoder = 4

Maximal number of generator batches = 2 number of training steps = 26,000

Validation every = 750 steps

Minimal summary length = 30

Maximal summary length = 30

I have trained my transformer encoder-decoder model with inshorts dataset for initializing the model. Firstly, let’s get idea about Inshort dataset. It has 55k news data with proper articles and headlines.
I have shuffled the dataset for gathering multiple news on a timeline for training purposes. Then cleaned the dataset as it has short terms and symbols which are unnecessary and could be a hindrance for training. After classifying the data, I have prepared the dataset vocabulary for abstraction and this vocabulary is used in the data at least minimum repeat times, and also made sure all the last non digit non alphabet chars are removed. Then calculated words for encoding and decoding as they are important for positional encoding. Assigned a number to each word in order to find it in word embeddings. Then splitted the data in in training and testing mode. After preprocessing the whole dataset, Defined the data model using scaled dot product function using multi headed attention classification which is used by TensorFlow. After that generating layers for positional encoding. For Embedding layers, I have used pre-trained GloVe embeddings, and developed the final model using keras transformers model. Then did a padding mask for masking "pad" sequences so it won’t lose while training. Also used Adam optimizer for more accurately trained dataset. Attached a graph of all my training and validation loss where I got 4.70 loss with 178 epochs.
To train this model for 15 Epochs its smallest validation loss is 4.80 which is still big and to train this model it took 18 hours. After training the model, all the results and outputs are mentioned in the results chapter.

To get the more accurate and less validation loss I tried to increase the epochs to 300 and it crashed due to less memory and GPU power mentioned earlier it needs heavy specification hardware. It was able to get the validation loss to 4.50 in 6 days for 178 epochs. The screenshot for those results is mentioned below.

4.2 Transformer Model validation training graph on 15 epochs

we can use more data or more regularization to avoid the overfitting
4.3 Transformer Model validation training graph on 178 epochs

After training the model, the results are mentioned in the results section with its ROUGE scores.

After getting success in this inshorts datasets I tried to run this model using CNN/ Dailymail datasets but due to hardware limitations it was not able to get trained and has given errors which are shown below:
4.4.1 Error while generating the model

Furtherly, I tried the Bert fine-tuned Model and preprocessed the cnn/dailymail dataset and tried to run the model but as per limited hardware resources and less tensor sizes it stopped in middle of training on my personal system (2GHz dual core, 16GB ram, 1.5gb GPU). So, I transformed my code to google colab which has a more powerful GPU still it was not able to run.
4.5.1 successful training model of BERT Model on 3 Epoches

4.5.2 BERT model training failure due to less GPU power

Finally, I have trained the Google PEGASUS pre-trained model with a dataset which has given pretty understandable outputs. This model is charged with recovering numerous full sentences that were deleted from documents during pre-training. An incomplete document serves as the
input for this pre-training, while the output is made up of the incomplete sentences concatenated together. In contrast to solely supervised systems, which frequently experience bottleneck issues, self-supervision allows you to construct as many examples as there are documents. It generated tokens for each word from the training dataset and summary part with an attention mask of 1. For comparison it matches with tokens of summaries and main news words and then it generates the sentence from that token for meaningful summary output.

4.6.1 Successful PEGASUS model
5. Results and Evaluation

5.1 Results

5.1.1 Transformer Outputs

Below mentioned output is for 178 epochs after that it failed due to hardware limitation, Then tried to run this model on a different system (Google Colab [36]) which has more specification and GPU of 15 gb.

```
smallest val loss: (178, 4.5043836)
Out[49]:
{'lmdb': 12': 0.1, 'd_out_rate': 0.1, 'num_layers': 4, 'd_model': 50, 'd_ff': 512, 'num_heads': 5, 'init_lr': 0.001}

we can use more data or more regularization to avoid the overfitting
```

```
In [50]:
print(clean_words(longview_val['long'][11]))
print()
print(summarize(clean_words(longview_val['long'][11])))
ranj player hospitalised after being hit on the head

a yearold <UNK> <UNK> <UNK> was caught in a <UNK> <UNK> <UNK> in the <UNK> village on saturday after he was cau
ght in a <UNK> <UNK> <UNK> <UNK> was caught in the <UNK> village in the village of the <UNK> district of the <UNK> <UNK>
was a <UNK> <UNK> in the village of the village in the village
```

```
In [51]:
print(clean_words(longview_val['long'][12]))
print()
print(summarize(clean_words(longview_val['long'][12])))

rbi to inject more cash into banks in march

the rbi on wednesday said that the banks will make old notes to the banks of the banks of india and india to raise fu
nds to the banks of the banks and banks are not able to share their cash and banks to get their respective banks and
banks are <UNK> to make cash payments and cash deposits
```

5.1.1.1 Transformer model outputs

After training the model on 277 epochs its loss reduced to 3.75 and has given more short and precise output which is mentioned below:
5.1.1.2 Output of Transformer Model with 277 epochs

That was an impressive comparison between the golden summary and output. In the field of Text Summarization, the standard measure for evaluating the accuracy of summarizer systems is based on summaries produced by humans, which are known as the Golden Set. However, instead of solely relying on this metric, it may be more effective to take a human-like approach in developing a summarizer system that can perform just as well as humans. By training a Neural Text Summarizer to learn from a human-style, it has the potential to become the most precise text summarizer.

5.1.2 Bert Model output

In experimentation, I conducted tests on the complete dataset and found that the BERT model surpassed the traditional transformer model, indicating its effective application on texts exceeding 512 tokens. However, the ultimate performance of the BERT model remained inferior to that of the model trained on truncated text. Despite this, the comparable outcome of the baselines suggests that the dataset's specific traits, such as the likelihood of important information being
contained in the opening sentences of a text, accounted for this variance. In summary, my findings indicate that while BERT is an optimal model, its performance may be influenced by the characteristics of the dataset being used.

5.1.2.1 BERT model outputs

<table>
<thead>
<tr>
<th>Reference</th>
<th>my dogs seem to like it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>dogs love it</td>
</tr>
<tr>
<td>Transformer</td>
<td>the dogs love it</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reference</th>
<th>meh</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>worst coffee ever</td>
</tr>
<tr>
<td>Transformer</td>
<td>worst coffee ever</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reference</th>
<th>great coffee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>a great coffee</td>
</tr>
<tr>
<td>Transformer</td>
<td>great everyday coffee</td>
</tr>
</tbody>
</table>

5.1.3 Pegasus output

While I tried to use the same input for CNN/daily mail dataset for below input:

input:

"In our Behind the Scenes series, CNN correspondents share their experiences in covering news and analyze the stories behind the events. Here, Soledad O'Brien takes users inside a jail where many of the inmates are mentally ill. An inmate housed on the "forgotten floor," where many mentally ill inmates are housed in Miami before trial. MIAMI, Florida (CNN)--The ninth floor of the Miami-Dade pretrial detention facility is dubbed the "forgotten floor." Here, inmates with the most severe mental illnesses are incarcerated until they're ready to appear in court. Most often, they face drug charges or charges of assaulting an officer, charges
that Judge Steven Leifman says are usually "voidable felonies". Others are housed in "the forgotten floor," the top floor of the pretrial facility that is also used by criminal offenders. What makes these inmates separate is their medical needs. They're given special medications to treat their mental illness, but it's only in those cases where necessary to take certain medications themselves for the very dangerous condition. These medications can lead to anaphylaxis or dangerous reactions if taken by people who are on them alone. A mentally ill person like John Brown, a convicted felon with severe mental illness. He spends four months in the "forgotten floor" and is charged with being a felon, disorderly conduct, assault with a deadly weapon (a gun), possession of child pornography, resisting arrest, and possession of marijuana. Brown was on the second floor when he fell asleep in bed. On April 11, his bed fell over and he was arrested and held until jail officials could check on him. After his detention, Brown refused to return to court the next morning to answer the charges. "I won't be back at all," Brown said. "There will be no new hearings or there will be no court." This episode will air on January 18, 2013 on CNN's Inside Story."

baseline: "In our Behind the Scenes series, CNN correspondents share their experiences in covering news and analyze the stories behind the events. Here, Soledad O'brien takes users inside a jail where many of the inmates are mentally ill. An inmate is housed on the "forgotten floor," where many mentally ill inmates are housed in Miami before trial. MIAMI, Florida (CNN) -- The ninth floor of the Miami-Dade pretrial detention facility is dubbed the "forgotten floor."

output: "CNN's Soledad O'brien takes us on a journey inside a Miami-Dade
pretrial detention facility where mentally ill inmates are housed before their trial. The ninth floor of the facility, known as the "forgotten floor," houses many of these inmates.”

5.1.3 PEGASUS model Output

5.1.4 Comparison on sample text:

For comparison purposes I used the same news paragraph in each model and checked its output which has given me different results and also checked with ROUGE scores that can evaluate its accuracy.

<table>
<thead>
<tr>
<th>Model</th>
<th>Sample Text</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformer Encoder-decoder</td>
<td>In our Behind the Scenes series, CNN correspondents share their experiences in covering news and analyze the stories behind the events. Here, Soledad O'Brien takes</td>
<td>CNN's Soledad O'brien takes us on a journey inside a Miami-Dade pretrial detention facility where mentally ill inmates are housed before their trial. The</td>
</tr>
</tbody>
</table>
users inside a jail where many of the inmates are mentally ill. "I won't be back at all," Brown said........  "There will be no new hearings or there will be no court." This episode will air on January 18, 2013 on CNN's Inside Story.

| BERT model | In our Behind the Scenes series, CNN correspondents share their experiences in covering news and analyze the stories behind the events. Here, Soledad O'Brien takes users inside a jail where many of the inmates are mentally ill. "I won't be back at all," Brown said.........  "There will be no new hearings or there will be no court." This episode will air on a journey inside a Miami-Dade pretrial detention facility where mentally ill inmates are housed before their trial. | CNN's Soledad O'Brien takes us on a journey inside a Miami-Dade pretrial detention facility where mentally ill inmates are housed before their trial. |
Pegasus Model

In our Behind the Scenes series, CNN correspondents share their experiences in covering news and analyze the stories behind the events. Here, Soledad O'Brien takes users inside a jail where many of the inmates are mentally ill. "I won't be back at all," Brown said...........

"There will be no new hearings or there will be no court." This episode will air on January 18, 2013 on CNN's Inside Story.

| In Miami-Dade's pretrial detention facility, mentally ill inmates are housed on the ninth floor, also referred to as the "forgotten floor." CNN's Soledad O'Brien provides an inside look at the facility and the conditions these inmates face before their trials |

### 5.2 Evaluation

This section evaluates the proposed summarizing models and hypotheses. The experimental analysis was conducted with two factors in mind: quality and quantity.

Evaluating the quality and readability of summary models can be difficult because the notion of
an ideal summary can differ from person to person. It's important to be noted that validation accuracy is not an appropriate statistic for summarization activities. Thus far, the Rouge score system has been utilized as the sole criteria to evaluate a summarizer model in several investigations.

ROUGE metrics

In the conducted experiments, the values of ROUGE-1, ROUGE-2, and ROUGE-L were determined for every summary, as stated in reference [32]. ROUGE-N (1, 2), as well as ROUGE-L, were used in the evaluation process.

ROUGE-N

ROUGE-N is the number of n-gram co-occurrences between candidate and reference summaries, where n is the number of words to match. We collect the ratio of the number of single words matched by the number of words in the reference summary in the 1-gram metrics. The ratio of two continuous words matching in both summaries is represented by 2-gram metrics. When summarizing multiple texts, the average of all the n-gram values is used[34].

\[
\text{ROUGE-N} = \frac{\sum_{S \in \text{ReferenceSummaries}} \sum_{\text{gram}_n \in S} \frac{\text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \text{ReferenceSummaries}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)}}{\sum_{\text{num.of documents}}} \]

\[
\text{ROUGE-N(Avg)} = \frac{\sum R_n}{\text{num.of documents}}
\]

Generally, evaluation measures are based on precision or recall. Precision is defined as the ratio
of matching words in candidate and reference summaries to the number of words in the candidate summary.

\[
\text{Precision} = \frac{\text{Num. of matching } n \text{-units}}{\text{Num. of words in candidate summary}}
\]

Recall-based models take into account the number of matched n-units divided by the number of n-grams in the reference summary.

\[
\text{Recall} = \frac{\text{Num. of matching } n \text{-units}}{\text{Num. of words in reference summary}}
\]

**ROUGE-L**

Rouge-L considers the longest common subsequence of terms matching between the candidate and reference summaries. One advantage of utilizing Rouge-L over n-gram ROUGE measures is that sentences with no continuous match words can still contribute to the summary score[34]. The n-gram does not need to be defined before starting the evaluation because it will select continuous n-units by default. The precision, recall, and F-measure for Rouge-L are as follows[34].

\[
P_{LCS} = \frac{LCS(C, R)}{\text{words}_{\text{candidate}}}
\]

\[
R_{LCS} = \frac{LCS(C, R)}{\text{words}_{\text{reference}}}
\]

Another metric based on both precision(p) and recall(r) known as F-measure(f) is evaluated as follows.

\[
F_{LCS} = \frac{(1 + \beta^2)R_{LCS} \times P_{LCS}}{R_{LCS} + \beta^2 \times P_{LCS}} \text{ where } \beta \leq 1
\]
In the CNN/dailymail Datasets implementation, two variations of ROUGE[38] are calculated: one calculates the score per sentence and averages it for the summaries (ROUGE-L), and the other calculates it directly over the whole summary (ROUGE-Lsum).

**5.3.1 Code for evaluating model**

Below Mentioned table is categorizing the ROUGE scores for all 3 models on CNN/Dailymail datasets:

<table>
<thead>
<tr>
<th>Model Name</th>
<th>ROUGE-1</th>
<th>ROUGE-2</th>
<th>ROUGE-L</th>
<th>ROUGE-Lsum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformer</td>
<td>25.39</td>
<td>8.73</td>
<td>23.15</td>
<td>28.57</td>
</tr>
<tr>
<td>BERT</td>
<td>38.25</td>
<td>18.09</td>
<td>36.45</td>
<td>39.76</td>
</tr>
<tr>
<td>PEGASUS</td>
<td>41.79</td>
<td>24.0</td>
<td>38.93</td>
<td>45.09</td>
</tr>
</tbody>
</table>

Quantitative Analysis: All three models used the identical data for calculating their Rouge
ratings, which are then compared. According to the results, Pegasus outperformed the other models for the given CNN/ Dailymail dataset.

5.3 Limitations of the Methods after experimentation

5.3.1 Transformer Model:

- The regular Transformer model has a limited ability to capture the context and semantics of the input text, which can result in the development of irrelevant or erroneous summaries.
- Trouble with long sequences: The CNN DailyMail dataset contains lengthy articles, and the vanilla Transformer model may struggle with such long input sequences. As a result, incomplete or erroneous summaries may be generated.
- Input text coverage is limited: The vanilla Transformer model can produce summaries that do not include all of the important information in the input text.
- As noted by Zou et al. (2021), one of the main limitations of the vanilla Transformer model is its struggle to capture the context and semantics of the input text, which can lead to the generation of irrelevant or inaccurate summaries

5.3.2 BERT Model:

- Restricted capacity to generate fresh text: The BERT model is a pre-trained model that has been fine-tuned for specific tasks. It has a limited ability to generate new text that is not included in the input text. This can result in the development of summaries that are similar to the input content.
- Difficulties in generating long summaries: The BERT model can generate brief and to-
the-point summaries, but it struggles to construct longer summaries that capture all of the significant information in the input text.

5.3.3 Pegasus Model

- Computationally expensive: The PEGASUS model is a huge model that necessitates a significant amount of processing power to train and deploy. This can be a problem for some applications.

- The PEGASUS model, like the vanilla Transformer model, can generate summaries that do not cover all of the important information in the original text. This can be a limitation for some applications that demand a full summary.

- Difficulties in producing coherent summaries: The PEGASUS model can produce fluent and grammatically correct summaries, but it struggles to provide cohesive summaries that capture the major ideas of the original text. This can be a hindrance in some situations where summary coherence is crucial.
6. Future Work

My analysis in this thesis revealed that each model had its own set of strengths and shortcomings, and that the model to be used would be determined by the application's specific requirements. In the future, various fields of research could be investigated. The development of more efficient and scalable methods for abstractive text summarization is one example. While the PEGASUS model is currently the most advanced, it requires a large amount of computational resources to train and execute that is also experienced while training it on a personal system, which may be prohibitively expensive for some applications (Zhang et al., 2021). Creating models that may produce comparable results with fewer resources would be a significant direction for future research.

Exploration of multi-modal summarization is another field for future research. While the CNN/DailyMail dataset is mostly made up of text-based articles and summaries, many real-world applications require a combination of text, graphics, and video. Creating models that can efficiently synthesize information from numerous modalities would be a significant topic for future research.

Finally, as the area of NLP advances, new models and techniques emerge at an increasing rate. Integration of generative language models, such as ChatGPT, into abstractive text summarization is one intriguing avenue for future research (Radford et al., 2021). While our investigation concentrated on three specific models, the area is continually changing, and new models and methodologies are expected to emerge in the next few years.

In the coming years, researchers will need to focus on developing more efficient and scalable models, investigating multi-modal summarization, and incorporating new models and methodologies.
7. Conclusion

In this thesis, we compared three distinct models for abstractive text summarization on the CNN/DailyMail dataset: the Transformer model, the BERT model, and the PEGASUS model.

My research revealed that all three models were capable of producing high-quality summaries, but each had its own set of advantages and disadvantages. While the Transformer model was a popular starting point, it struggled to capture the context and semantics of the input text, which could result in inaccurate or irrelevant summaries. The BERT model, on the other hand, was better at capturing the semantics of the input text but struggled to generate diverse and interesting summaries. The PEGASUS model, which was specifically created for abstractive text summary, was capable of producing extremely informative and diverse summaries, but training and running it took a significant amount of computer resources.

For the CNN/DailyMail dataset, the PEGASUS model beat the vanilla Transformer and BERT models in terms of both ROUGE scores and human evaluation measures. This shows that for abstractive text summarization on this dataset, the PEGASUS model may be the best option. However, the computational requirements for the PEGASUS model are much higher than those for the vanilla Transformer or BERT models, which may be a factor in some applications.

Overall, our analysis of the three models reveals that on the CNN/DailyMail dataset, there is no one-size-fits-all approach for abstractive text summarization. The model used will be determined by the application's specific requirements as well as the resources available for training and executing the model. The PEGASUS approach, on the other hand, appears to be a potential option for applications that demand high-quality, diversified, and interesting summaries.
8. References


cnn_dailymail · Datasets at Hugging Face, https://huggingface.co/datasets/cnn_dailymail.


from https://www.connectedpapers.com/main/f4061bd225b3be5b3f5b18eb1a229ce991efefeb/PEGASUS%3A-
Pre%20training-with-Extracted-Gap%20sentences-for-Abstractive-
Summarization/graph

Medium. Retrieved March 28, 2023, from https://towardsdatascience.com/bert-explained-
state-of-the-art-language-model-for-nlp-f8b21a9b6270

Ontology-aware pretrained language model for end-to-end task-oriented dialogue. MIT
Press. Retrieved April 11, 2023, from
https://direct.mit.edu/tacl/article/doi/10.1162/tacl_a_00534/114595/OPAL-
Ontology-Aware-Pretrained-Language-Model-for


[26] Pegasus: Pre-training with extracted gap-sentences for abstractive ... (n.d.). Retrieved

[27] Ramesh Nallapati et al. “Abstractive Text Summarization Using Sequence-to-Sequence
RNNs and Beyond”. In: (2016) (cit. on pp. 2, 3, 7, 9, 12, 14, 16, 20, 30, 31).

details, just the summary! topic-aware convolutional neural networks for extreme
https://arxiv.org/abs/1808.08745


[34] Sjobergh, J. (2007). Older versions of the rogue eval sum-¨ summarization evaluation system were easier to fool. Information Processing & Management, 43(6):1500 – 1505.


48
https://doi.org/10.1016/j.jpsychires.2019.04.007

