MOHAMMED KHALID HILMI BRIMAN

ATILIM UNIVERSITY 2023

A COMPREHENSIVE EVALUATION METRIC FOR ABSTRACTIVE SUMMARIZATION: LEVERAGING SIMILARITY, ENTAILMENT, AND ACCEPTABILITY

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

OF

ATILIM UNIVERSITY

MOHAMMED KHALID HILMI BRIMAN

A MASTER OF SCIENCE THESIS

IN

THE DEPARTMENT OF COMPUTER ENGINEERING

MAY 2023

A COMPREHENSIVE EVALUATION METRIC FOR ABSTRACTIVE SUMMARIZATION: LEVERAGING SIMILARITY, ENTAILMENT, AND ACCEPTABILITY

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

BY

MOHAMMED KHALID HILMI BRIMAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR

THE DEGREE OF MASTER OF SCIENCE

IN

COMPUTER ENGINEERING

MAY 2023

Approval of the Graduate School of Natural and Applied Sciences, Atilim University.

_____

Prof. Dr. Ender KESKİNKILIÇ
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of **Master of Science in Computer Engineering Department, Atilim University**.

_____

Assoc. Prof. Dr. Gökhan ŞENGÜL
Head of Department

This is to certify that we have read the thesis **"A COMPREHENSIVE EVALUATION METRIC FOR ABSTRACTIVE SUMMARIZATION: LEVERAGING SIMILARITY, ENTAILMENT, AND ACCEPTABILITY"** submitted by **MOHAMMED KHALID HILMI BRIMAN** and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Assoc. Prof. Dr. Beytullah YILDIZ
Supervisor

Examining Committee Members:

Prof. Dr. Mehmet S. AKTAŞ
Computer Eng. Department,
Yildiz Teknik University _____

Assoc. Prof. Dr. Beytullah YILDIZ
Software Eng. Department, Atilim University _____

Assoc. Prof. Dr. Çiğdem TURHAN
Software Eng. Department, Atilim University _____

Date: May 26, 2023

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Mohammed Khalid Hilmi Briman

Signature:

ABSTRACT

A COMPREHENSIVE EVALUATION METRIC FOR ABSTRACTIVE SUMMARIZATION: LEVERAGING SIMILARITY, ENTAILMENT, AND ACCEPTABILITY

Briman, Mohammed Khalid Hilmi
M.S., Department of Computer Engineering
Supervisor: Assoc. Prof. Dr. Beytullah YILDIZ

May 2023, 106 pages

Producing meaningful automatic summaries from long textual documents is essential in various fields. The emergence of novel neural network architectures, such as the Transformer model, has led to the development of large pre-trained language models that can produce quality summaries. However, model-generated summaries suffer from many issues. Thus, standard automatic evaluation metrics, such as the ROUGE metric, fail to effectively evaluate the quality of summarization models. In this study, we introduce SEAScore, a new model-based automatic evaluation metric that can evaluate model-generated summaries against their counterpart reference summaries by utilizing multiple Natural Language Processing tasks such as Semantic Similarity, Natural Language Inference, and Linguistic Acceptability. SEAScore takes features extracted by pre-trained language models and produces an evaluation score to measure the quality of summarization models. In this thesis, we develop our new evaluation metric SEAScore and train three summarization models to assess our new metric. Experimental results show that SEAScore correlates better with human judgment than some standard metrics.

Keywords: Abstractive Text Summarization, Natural Language Processing, Semantic Similarity, Natural Language Inference, Linguistic Acceptability, Transformer, Automatic Evaluation Metric.

ÖZ

SOYUTLAYICI ÖZETLEMEK, BENZERLİK, GEREKLİLİK, VE KABUL EDİLEBİLİRLİĞİ KULLANAN KAPSAMLI DEĞERLENDİRME METRİĞİ

Briman, Mohammed Khalid Hilmi
Yüksek Lisans, Bilgisayar Mühendisliği
Tez Yöneticisi: Doç. Dr. Beytullah YILDIZ

Mayıs 2023, 106 sayfa

Uzun metinlerden otomatik olarak anlamlı özetler üretmek, birçok alanda büyük önem taşımaktadır. Transformer modeli gibi yeni sinir ağı mimarilerinin ortaya çıkması, kaliteli özetler üretebilen çok sayıda büyük dil modellerinin gelişmesine neden olmuştur. Fakat, özetleme modellerinin ürettiği özetler, önemli bir sorunu beraberinde getirmektedir. Özetleme modellerinin kalitesini ölçen, ROUGE gibi, standart otomatik değerlendirme metrikleri, kapsamlı bir değerlendirme yapmakta eksik kalmaktadır. Bu çalışmada, modeller tarafından üretilen ve insanlar tarafından yazılan örnek özetleri kullanan, SEAScore adlı yeni bir model tabanlı metrik sunuyoruz. Bu metrik, semantik benzerlik, doğal dil çıkarımı ve dilsel kabul edilebilirlik gibi çeşitli Doğal Dil İşleme yöntemlerini kullanır. Geliştirdiğimiz SEAScore metriği, daha önce eğitilmiş dil modelleri tarafından çıkarılan özellikleri kullanarak, özetleme modellerinin kalitelerini ölçen bir puan üretir. Bu tezde, üç tane özetleme modeli kullanarak yeni metriğimizin kalitesini ölçen deneyler yaptık. Deneysel sonuçlara göre, geliştirdiğimiz SEAScore metriği, bilinen standart metriklerine göre, insan tarafından üretilen değerlendirme puanları ile daha yüksek korelasyon sergileyerek başarılı sonuçlar sunmuştur.

Anahtar Kelimeler: Soyutlayıcı Metin Özetleme, Doğal Dil İşlemi, Semantik Benzerlik, Doğal Dil Çıkarımı, Dilsel Kabul Edilebilirlik, Transformer, Otomatik Değerlendirme Metriği

To my family

ACKNOWLEDGMENTS

I would like to thank my professor for guiding me and giving valuable feedback to push the research forward.

I want to thank my parents and sister for supporting me throughout my graduate studies.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF SYMBOLS/ABBREVIATIONS

NLP	Natural Language Processing
STS	Semantic Textual Similarity
NLI	Natural Language Inference
MNLI	Multi-Genre Natural Language Processing
CoLA	Corpus of Linguistic Acceptability
RNN	Recurrent Neural Network
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
BiLSTM	Bidirectional Long Short-Term Memory
GPU	Graphics Processing Units
Seq2seq	Sequence-to-sequence
GRU	Gated Recurrent Unit
RL	Reinforcement Learning
BERT	Bidirectional Encoder Representation from Transformer
MLM	Masked Language Model
PLM	Permuted Language Model
RoBERTa	Robustly Optimized BERT Pre-training Approach
DeBERTa	Decoding-enhanced BERT with Disentangled Attention
MPNet	Masked and Permuted Pre-training for Language Understanding
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
BLEU	Bilingual Evaluation Understudy
SEAScore	Similarity, Entailment, and Acceptability Score
T5	Text-to-Text Transfer Transformer
BART	Bidirectional Auto-Regressive Transformer
HF	HuggingFace
TF	TensforFlow
PT	PyTorch
LLM	Large Language Model

CHAPTER 1

INTRODUCTION

1.1 Overview

With the exponential increase of textual information from various platforms on the internet and the impressive advancements in the field of Natural Language Processing, extracting valuable and meaningful information from long textual data has become crucial. Automatic text summarization is an NLP task that generates summaries containing the most important and relevant information from source documents [1]. The importance of text summarization lies in enabling readers to quickly understand the content of a document, or a set of documents, without the need to dive into the details of those documents. There are many human-made examples of summarizing long documents that take many forms. For example, news journals use their headlines by creating a headline that encompasses the content of their articles. Similarly, news journals often present bullet points summarizing the article's main points. Another example is the use of abstracts in scientific journal articles that summarize essential parts of the journal in a single paragraph without diving into details. There are two types of automatic text summarization approaches, extractive and abstractive summarization. Extractive summarization generates a summary using words and sentences from the source document [1]. Conversely, abstractive summarization produces a summary using entirely new words that may not exist in the source document [1], which is a much more difficult task compared to extractive summarization since the abstractive summary must be understandable and grammatically correct.

There have been several approaches for abstractive text summarization models; one of the most prominent architectures used is an RNN encoder-decoder model in which an encoder reads input data, stores it into a latent vector, and feeds it into a decoder that

uses that data to generate new sequences of textual data as an output [2]. Moreover, the emergence of the attention mechanism by Bahdanau et al. [3] led to significant improvements in text generation models by allowing them to generate longer text sequences [3]. Thus, the attention mechanism enhanced the performance of text summarization models. The combination of encoder-decoder models and the attention mechanism enabled several breakthroughs in the field of NLP and abstractive text summarization by introducing state-of-the-art sequence-to-sequence (seq2seq) summarization models [2], [4], [5]. In addition, Vaswani et al. [6] introduced the Transformer architecture, a novel and simple neural network type that depends entirely on the attention mechanism and removes the need for recurrence. The Transformer model proved to perform significantly better than traditional RNNs [2] and LSTMs [7], allowing for faster training of text generation models [6].

The Transformer model changed the landscape of the NLP field as it allowed researchers to experiment with novel Transformer-based models to improve the text summarization task and text generation tasks in general; this resulted in the emergence of large Transformer-based language models (LLMs) that are pre-trained on vast textual corpora, and these models can then be finetuned on downstream tasks such as abstractive text summarization. LLMs such as GPT-2 [8], GPT-3 [9], BLOOM [10], T5 [11], and BART [12] achieved state-of-the-art results in multiple text generation tasks, including abstractive text summarization [8], [11], [12], [13]. The main advantage of these models is the knowledge they gained from billions of textual data, allowing them to generate meaningful text and better summaries accurately [8], [11], [12], [13].

Although state-of-the-art summarization models have the ability to produce quality summaries, they still suffer from many linguistic problems, such as grammatical errors, logical agreement problems, and the addition of irrelevant information in the generated summaries [14]. Thus, effective evaluation of the quality of generated summaries is crucial. These evaluations can be conducted by humans. However, this can be cumbersome and time-consuming. So, automatic evaluation metrics are used to

measure the quality of a summary produced by a summarization system. The most widely used metric for text summarization is the ROUGE [15] metric, which compares a model-generated summary against a reference summary provided by humans by counting the overlapping n-grams between the two summaries [15]. Several other popular metrics like BLEU [16] and METEOR [17], follow a similar method to ROUGE. Although these metrics are popular, they fail to capture contextual information between the words in the summaries. Most recently, a novel metric was proposed by Zhang et al. [18] called BERTScore, which utilizes BERT's [19] ability to produce contextualized embeddings for candidate and reference summaries.

The emergence of large pre-trained language models, such as BERT [19] and its variants, has opened a new area of research for new evaluation metrics that harness these models' power, allowing for a more accurate evaluation of summarization models. This study aims to propose a new evaluation metric that uses the abilities of three large pre-trained language models (LLMs) to help extract linguistic features from candidate and reference summaries and produce a final evaluation score that reflects the quality of a candidate summary compared to its counterpart reference summary. The multitude of linguistic features allows our metric to measure the quality of a summary by utilizing three specific NLP tasks, including semantic textual similarity (STS), linguistic acceptability, and natural language inference (NLI). Thus, this will give a more comprehensive picture of how to assess summarization systems. Our contribution to this study can be summarized as follows:

1. Proposing a new automatic evaluation metric called SEAScore (Similarity, Entailment, and Acceptability Score) which uses several pre-trained language models.

2. Training a Transformer model from scratch and using transfer learning to finetune two pre-trained language models on the abstractive summarization task.

3. Conducting experiments that compare the results of our metric to widely used standard evaluation metrics.

This thesis is organized as follows; Chapter 2 will present a literature review that discusses past abstractive summarization models and automatic evaluation metrics. Chapter 3 will go into the details of the tools, models, and metrics that will be used throughout the thesis work. Chapter 4 will present the methodologies used to develop our new metric and provide the details of the training process of our summarization models. The results of the experimentation will be provided in Chapter 5. Finally, in Chapter 6, we will present a conclusion of our work and discuss the potential future directions of the work proposed by this thesis.

CHAPTER 2

RELATED WORK

2.1 Models and Metrics for Abstractive Summarization

In this section, a literature review is conducted to investigate different models, techniques, and metrics used in the field of text summarization.

2.1.1 Abstractive Summarization Models

Rush et al. [4] from Facebook artificial intelligence research lab proposed a sentence-level, attention-based abstractive text summarization model that uses generation algorithms. The success of neural machine translation models inspired this model. It used a Bahdanau et al. [3] attention-based encoder and a beam-search decoder [4]. The experiments used the DUC 2004 [20] evaluation and Gigaword datasets [4].

Nallapati et al. [2] introduced an abstractive text summarization model that uses a seq2seq RNN architecture with attention. This model aimed to solve critical problems in abstractive summarization that previous architectures failed to address. The model consisted of a bidirectional GRU-RNN encoder and a unidirectional GRU-RNN decoder [2]. Lastly, the authors used the CNN/DailyMail [21] dataset to conduct their experiments.

See et al. [5] proposed a novel architecture called a Pointer-Generator network that is able to accurately reproduce information from the source document via pointing while keeping the ability to produce new words by using the generator [5]. Moreover, the Pointer-Generator network uses a coverage mechanism to keep track of the information that has been summarized to avoid any repetition in the generated

summaries [5]. Experiments were conducted on the CNN/DailyMail [21] dataset with significant improvements compared to previous models [5].

Chen et al. [22] proposed a summarization method that mimics how humans summarize long documents by selecting relevant sentences and then abstractively rewriting those sentences [22]. This model combines extractive and abstractive summarization by introducing a hybrid extractive-abstractive model with an RL model based on policy; the RL model acts as a bridge that combines the two networks [22]. This novel method made the model aware of the word hierarchy in sentences [22]. The model archived state-of-the-art results when experimented on the CNN/DailyMail dataset [21].

Gehrmann et al. [23] sought to introduce a bottom-up approach to abstractive text summarization that focused on the content selection aspect of summarization [23]. The paper used a simple two-step technique: a content-selector selects the most important phrases from the source document that should be a part of the generated summary and then constrains the model to select the most likely phrases [23].

Kryscinski et al. [24] aimed to improve the level of abstraction in summarization models. The paper proposed an encoder-decoder model in which the decoder part is turned into a contextual network that retrieves the most relevant parts of the source document [24]. The decoder also contained a pre-trained language model with prior text generation knowledge [24]. The paper also applied a custom metric that used policy learning to encourage generating new phrases [24]. The encoder used a bidirectional LSTM (or a BiLSTM) [25], and the decoder used a 3-layer LSTM architecture [7]. The proposed model achieved state-of-the-art results on the CNN/DailyMail dataset [21].

Vaswani et al. [6] introduced a ground-breaking novel architecture that paved the way for a wide range of research areas in NLP and deep learning in general. The novel Transformer architecture aimed to remove the recurrence method altogether and rely

entirely on the attention mechanism. By using attention, it can map dependencies between inputs and outputs [6]. The Transformer adopted an encoder-decoder architecture that used a novel multi-headed attention mechanism called Self-attention [6]. The novelty of Transformer introduced a wide array of new pre-trained models that achieved state-of-the-art results in several NLP tasks, including text summarization. Moreover, experimental results significantly improved machine translation and English constituency tasks [6].

Raffel et al. [11] explored the idea of using transfer learning and turning every text-based NLP problem into a text-to-text problem, meaning that both the input and output are texts [11]. The idea here was to pre-train a model on a large corpus before fine-tuning that model on downstream tasks [11]. Raffel et al. [11] proposed the T5 model that used a Vaswani et al. [6] Transformer architecture and a custom dataset, C4, for pre-training the T5 model [11]. The proposed model achieved state-of-the-art results in text summarization on the CNN/DailyMail [21] dataset.

Lewis et al. [12] introduced a denoising autoencoder called BART, which used bidirectional and autoencoding Transformers [12]. The model was used for pretraining seq2seq models where the pretraining process had two steps: firstly, the model used a noising function to corrupt the text, then, a seq2seq model learned how to reconstruct the corrupted text to its original form [12]. BART is highly effective for text generation tasks and has achieved state-of-the-art results in text summarization on the XSUM [26] and CNN/DailyMail [21] datasets.

Liu et al. [27] aimed to use BERT [19] to propose a general-purpose framework for both abstractive and extractive text summarization tasks [27]. This method used a BERT-based encoder to obtain semantics from source documents and find representations from their sentences [27]. Regarding abstractive summarization, the paper proposed a model consisting of a pre-trained BERT encoder and a Transformer-based [6] decoder that is randomly initialized [27]. Experiments were conducted on the CNN/dailyMail [21] and the XSUM datasets [26].

Zhang et al. [13] introduced a Transformer-based [6] encoder-decoder model called PEGASUS. The model is pre-trained on large text corpora and uses a novel self-supervised objective [13]. Moreover, the PEGASUS encoder-decoder model applied BERT's [19] Masked Language Model (MLM) and Gap Sentence Generation (GSG) methods in its architecture [13]. Experiments for this model were conducted on several summarization datasets, including CNN/DailyMail [21] and XSUM [26].

2.1.2 Results for Abstractive Text Summarization Models

The following table summarizes the ROUGE-1, ROUGE-2, and ROUGE-L metric results from the summarization models mentioned in section 2.1.1 along with their respective datasets.

Table 2.1 ROUGE scores for summarization models with datasets

Model	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**Datasets**
ABS+ [4]	28.18	8.49	23.18	DUC-2004
ABS+ [4]	31.00	12.65	28.34	Gigaword
words-lvt2k-temp-att [2]	35.46	13.30	32.65	CNN/DailyMail
pointer-generator + coverage [5]	39.53	17.28	36.38	CNN/DailyMail
rnn-ext + abs + RL + rerank [22]	40.88	17.80	38.54	CNN/DailyMail
Bottom-Up [23]	41.22	18.68	38.34	CNN/DailyMail
ML+RL ROUGE+Novel, with LM [24]	40.19	17.38	37.52	CNN/DailyMail
T5-11B [11]	43.52	21.55	40.69	CNN/DailyMail
BART [12]	44.16	21.28	40.90	CNN/DailyMail
BART [12]	45.14	22.27	37.25	XSUM
BERTSUMABS [27]	41.72	19.39	38.76	CNN/DailyMails
BERTSUMABS [27]	38.76	16.33	31.15	XSUM
$PEGASUS_{LARGE}(C4)$ [13]	43.90	21.20	40.76	CNN/DailyMail
$PEGASUS_{LARGE}(C4)$ [13]	45.20	22.06	36.99	XSUM

2.1.3 Evaluation Metrics

ROUGE [15] is the standard metric for evaluating text summarization systems. It evaluates the quality of a model-generated summary by comparing it to a human-generated reference summary. ROUGE relies on counting the number of overlapping units, such as n-grams, between a candidate summary and a reference summary [15]. Types of ROUGE metrics include ROUGE-1, ROUGE-2, and ROUGE-L [15]. The most significant disadvantage of ROUGE is relying solely on n-gram overlaps, which is not enough to determine the quality of a summary.

BLEU [16] is a precision-based evaluation metric that evaluates machine translation systems. However, it is also used to evaluate text summarization systems. The main difference between BLEU [16] and ROUGE [15] is that BLEU is a precision-based metric while ROUGE is a recall-based metric. Like ROUGE [15], BLEU relies on overlapping n-grams between candidate and reference summaries. However, BLEU's reliance on overlapping n-grams hinders its ability to evaluate a candidate summary accurately. Additionally, BLEU uses a brevity score to penalize generated summaries that are too short compared to reference summaries [16].

METEOR [17] is another widely used evaluation metric for tasks such as text summarization and machine translation. It uses unigram matching between candidate and reference summaries [17]. Compared to BLEU [16] and ROUGE [15] metrics, which use precision and recall, respectively, METEOR calculates both unigram-recall and unigram-precision [17]. Moreover, a fragmentation measure is used to capture the quality of ordering of matching words in candidate summaries in relation to reference summaries [17].

Zhang et al. [18] introduced a novel text-generation evaluation metric called BERTScore [18]. BERTScore used a BERT-based model to obtain contextualized embeddings for a candidate and a reference summary, then it calculated a pairwise cosine similarity score for tokens in the candidate summary with tokens in the

reference summary [18]. BERTScore computed recall, precision, and F1 scores using the maximum similarity scores obtained from a greedy matching method [18]. This metric has been used to evaluate abstractive text summarization research by Gabriel et al. [28].

Zhao et al. [29] proposed a novel metric that used contextualized embeddings with an Earth Mover Distance measure. The metric is named MoverScore; it obtains contextualized embeddings for a candidate and reference summary and calculates the semantic distance between these summaries using a case of Earth Mover Distance called Word Mover Distance [30]. Compared to BERTScore [18], MoverScore [29] uses a method called soft alignment, a many-to-one relationship between tokens, while BERTScore uses hard alignment, a one-to-one relationship.

Kane et al. [31] proposed a novel architecture for a text generation metric called NUBIA. Instead of a mathematical architecture, NUBIA used a full model-based architecture consisting of three Transformer models and a feed-forward neural network aggregator [31]. NUBIA uses the neural features obtained from the pre-trained Transformer models. It feeds them into the aggregator to get a quality score that measures the quality of a candidate sentence compared to a reference sentence. The aggregator neural network model is trained on a machine translation dataset; there is insufficient information to determine how well it measures text summarization.

Peyrard et al. [32] suggested an evaluation metric using existing features from standard metrics like ROUGE [15] and human judgment scores. The metric takes a set of source documents, reference summaries, and a model-generated summary as inputs, and it tries to maximize a summary-level correlation with a regression framework [32]. The aim was to predict evaluation scores close to human judgment scores.

Clark et al. [33] proposed the Sentence Mover Similarity (SMS) metric that used both word and sentence embeddings. The SMS metric aimed to explore the effects of adding sentence embeddings to a metric. SMS implemented Word Mover Distance (WMD)

formula [30] to compute the minimum cost of "*moving"* both sentences and word embeddings from one document to embeddings in another one. The metric exhibited a consistent gain over ROUGE-L [15].

Scialom et al. [34] introduced a reference-free metric that applied a Reinforcement learning (RL) technique. The proposed metric, named SummaQA, eliminated the need for a human-generated reference summary and relied on a Question-Answering metric as a reward in an RL setup. This work relied on the idea that a good model-generated summary should contain relevant information from the source document by answering relevant questions extracted from the source document [34]. A drawback of this approach is the massive number of questions that can be generated for a single source document. Hence, this issue may harm its reproducibility [35].

Vasilyev et al. [35] proposed a metric to determine how useful a model-generated summary is to its readers. The proposed metric, named BLANC, used a BERT model [19] as an independent model that used a candidate summary to attempt to understand a source document. BLANC is also considered a reference-free metric since it only needs a model-generated summary and the source document to evaluate it [35].

Gao et al. [36] proposed SUPERT, which is described as an unsupervised method to evaluate multi-document summarization. Similar to SummaQA [34] and BLANC [35], SUPERT is also a reference-free metric that relies on extracting salient sentences from multiple documents to form a *pseudo-reference summary* [36]. The aim was to inspect the amount of information shared between the *pseudo-reference and candidate summaries* by computing a semantic similarity method [36].

Yuan et al. [37] utilized the performance of BART [12] to propose BARTScore, a metric that uses BART as its backbone pre-trained model. BARTScore uses a weighted log probability on a text given another text by utilizing the abilities of BART [12]. BARTScore aimed to treat the evaluation process as a text-generation problem. The

metric evaluates a text generation system from different perspectives, like faithfulness and fluency of the model-generated text [37].

CHAPTER 3

BACKGROUND

This chapter discusses the fundamentals of abstractive text summarization and other tasks by reviewing the foundations of all the tools and methods used throughout our research. We will go through concepts, models, and evaluation metrics.

3.1 Text Summarization

With the unprecedented explosion of textural information on the internet, the ability to quickly obtain valuable information has grown in significance, highlighting the importance of producing factual summaries that give the whole meaning of a document. Text summarization produces a significantly shorter version of a long document while trying to retain the core information of the source document. However, summarization is extremely difficult and cumbersome, so automating the summarization task, evaluating, and improving the quality of the generated summaries have gained significant traction as a research field.

With the growing advancements in the NLP field, many novel tools and methods have helped improve the summarization task. Nevertheless, producing factually consistent summaries remains to pose numerous challenges for researchers. There are many aspects to consider when summarizing a document, such as grammatical correctness, factual consistency, the phrasing of the summaries, and the semantic relations between words. Thus, evaluating generated summaries and their correlation with human-produced summaries is a crucial aspect to consider.

3.2 Summarization Methods

Automatic text summarization has two main approaches: Extractive text summarization and abstractive text summarization.

3.2.1 Extractive Text Summarization

Extractive text summarization uses key sentences from the source document to generate a summary. It chooses a subset of important sentences from a source text and combines them to form a summary that contains essential information from the source document [1]. Thus, as the name suggests, the method relies on extracting important sentences rather than generating original sentences. Since the summaries are made up of sentences from the source document, they tend to be grammatically correct and do not require further linguistic analysis.

An extractive summarization system contains the following components:

1. An extractive system forms an intermediate representation of the source input document and identifies the most important sentences from this representation [1].

2. Once this representation is obtained, the system assigns an "*importance"* score representing how well the intermediate representation captures some important information from the source text [1].

3. Finally, the system selects the *top-k* most significant sentences to generate a summary [1].

3.2.2 Abstractive Text Summarization

Abstractive text summarization systems generate summaries using original words and sentences that may not exist in the source document. This method starkly contrasts extractive systems and is much similar to how humans write summaries. Abstractive systems use advanced NLP techniques to produce summaries that contain the most important information from the source document without using sentences from the source text. The success of neural-based machine translation models inspired the emergence of abstractive text summarization models [4]. For example, Rush et al. [4] introduced a seq2seq encoder-decoder attention-based abstractive summarization model [4]. An example output of the mentioned model is given in Figure 3.2. Furthermore, abstractive summarization methods are more complex than extractive methods. They require more linguistic analysis to evaluate the quality of the summaries because abstractive summaries tend to suffer from factual consistency problems [14].

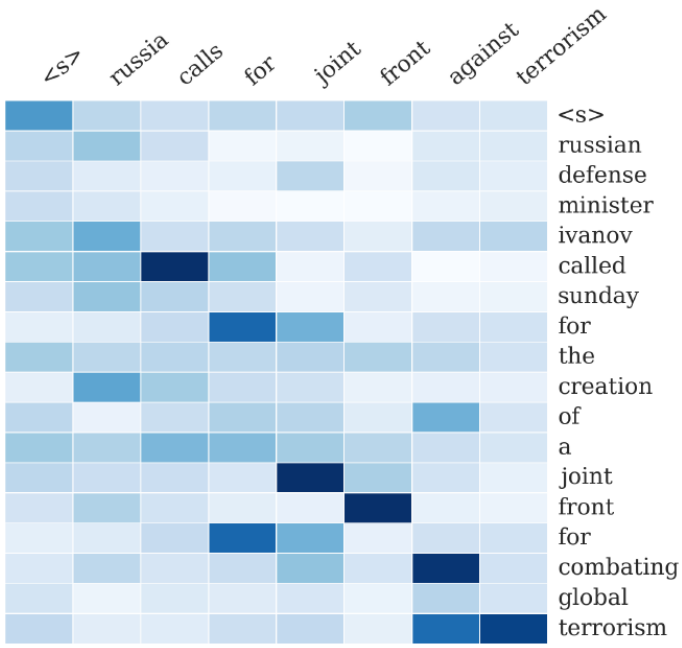

Figure 3.2 Encoder-decoder with attention heatmap graph example output [4]

3.3 Semantic Textual Similarity (STS)

Semantic Textual Similarity (STS) is a branch of NLP that measures the similarity between two or more pieces of text. STS task plays an important role in multiple NLP tasks such as text summarization, classification, and machine translation [38]. The earliest methods of measuring semantic similarity relied on text snippets that contained similar words or characters and techniques like Bag of Words (BoW) and Term Frequency – Inverse Document Frequency (TF-IDF) to represent text as real value vectors [38]. However, these techniques failed to consider that words often give different meanings depending on the context of the sentences.

Several types of semantic similarity methods incorporate a wide variety of techniques:

- **Knowledge-based semantic similarity:** This method uses knowledge-based methods and sources like lexical databases to compute the semantic similarity

between two terms. The knowledge source provides the context of those terms to remove ambiguous measures [38].

- **Corpus-based semantic similarity:** The semantic similarity calculation is done with the help of information provided and retrieved by large corpora [38]. This method also uses word embedding models like Word2Vec [39].

- **Deep neural network-based semantic similarity:** A wide variety of deep neural network types such as CNN, RNN, LSTM, and, most recently, the Transformer model [6] are used to create state-of-the-art semantic similarity models [38].

Figure 3.3 gives a comprehensive picture of all the STS method types.

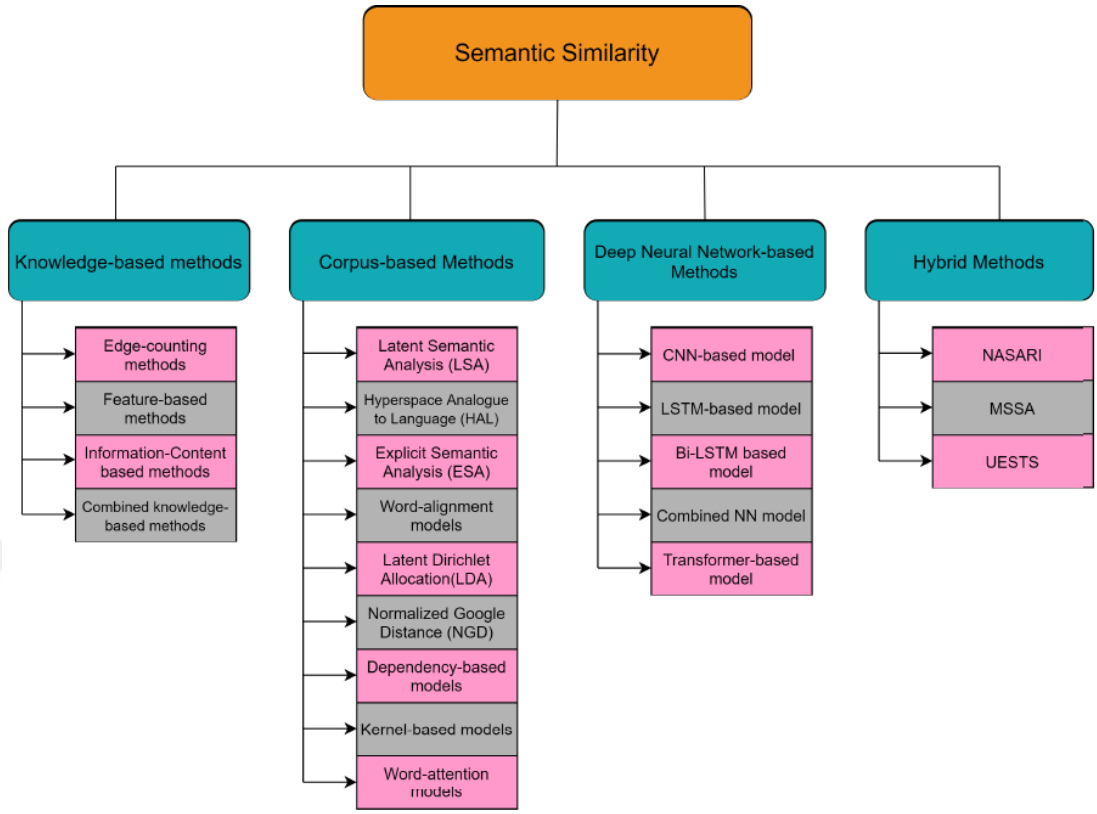

Figure 3.3 Semantic Similarity types and methods [38]

3.4 Natural Language Inference (NLI)

The Natural Language Inference (NLI) task is an NLP task that measures the semantic relations between two pieces of text [40]. This task aims to enable language models to achieve a human-level understanding of language [40]. An NLI model takes two sentences, a reference summary, and a candidate summary, and applies a classification task. The NLI classification task has three labels: *Entailment*, *contradiction*, and *neutral*. These classes correspond to 'True, 'False, and 'Undetermined,' respectively. The three labels mentioned before offer a broader understanding of the relationship between the two inputs by detecting any contradicting information, the addition of new information, or logical agreement between the texts. Table 3.4 demonstrates the NLI task with examples using the three labels.

Table 3.4 NLI task example

Premise sentence	**Hypothesis sentence**	**Label**
The dinner was delicious.	The dinner was terrible.	**Contradiction**
Two people were smiling at each other at the park.	Two women were looking at the dogs playing in the park.	**Neutral**
A football game with multiple males playing on Saturday.	Some men are playing a sport on Saturday.	**Entailment**

3.5 Models and Methods

The emergence of novel neural network architectures has significantly improved state-of-the-art results for multiple NLP tasks, including text summarization, semantic similarity, natural language inference, and sentence acceptability. The most important model that has paved the way for a plethora of research in NLP is the Transformer model [6] and its multiple applications.

3.5.1 Transformer

Introduced by Vaswani et al. [6], the Transformer is a neural network architecture that avoids recurrence and relies entirely on the attention mechanism [6]. Recurrent models require a previous state h_{t-1} to measure the current state h_t, this sequential nature prevents the parallelization of input sequences and fails to utilize the processing power of GPUs fully. Moreover, traditional recurrence models face a vanishing gradient problem with long sequences as they fail to capture important long dependencies. Since the Transformer model has entirely removed recurrence, it only uses the attention mechanism with feedforward networks to map dependencies between inputs and outputs. The Transformer model applies a self-attention mechanism that computes a “Scaled Dot-Product Attention” [6]. Self-attention is a type of attention that replaces

each element in a sequence with a weighted average of the remaining sequence [11]. The mechanism is defined as follows:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (3.5.1)$$

Where Q is a matrix that contains a set of queries, and K, V matrices hold keys and values respectively. Additionally, scaling is achieved using the term $\sqrt{d_k}$, where d_k is the dimension for input K. All three outputs, namely, Q, K, V are outputs of the preceding layers in the encoder-decoder architecture that the Transformer adopts.

The Transformer model adopts a Multi-Head Attention method. The Scaled Dot-Product is parallelized on multiple linear projections of the quires, keys, and values to perform the attention mechanism in parallel rather than computing a single attention function [6]. The outputs of these computations are then concatenated and projected again, yielding the final results that are then fed to a feedforward network. Moreover, both the encoder and decoder of the Transformer model share a similar structure. However, the decoder uses a different Multi-Head Attention where it applies masking to prevent the decoder from cheating when the model is training, which allows for a better next-token prediction. Finally, due to the absence of recurrence in the Transformer model, the authors added a positional encoding mechanism to the input sequence so that the model can understand the relative positions of the input tokens. There are many types of positional encodings; the authors implemented this method using *sine* and *cosine*, as illustrated in the following equations, where the position is denoted as pos, the dimension as i , and the output dimension as d_{model}:

$$PE_{(pos,2i)} = sin\left(pos/10000^{2i/d_{model}}\right) \quad (3.5.2)$$

$$PE_{(pos,2i+1)} = cos\left(pos/10000^{2i/d_{model}}\right) \quad (3.5.3)$$

The positional encoding method by Vaswani et al. [6] encodes the word positions by adding a vector that contains values in the range $[-1, 1]$ to the word embeddings. It achieves this using the equation's *sine* and *cosine* functions [41].

The following figure (Figure 3.5.1) illustrates the Transformer model with its encoder-decoder architecture, where the model takes a positionally encoded-word embedding input and an output that is forwarded to the decoder layer.

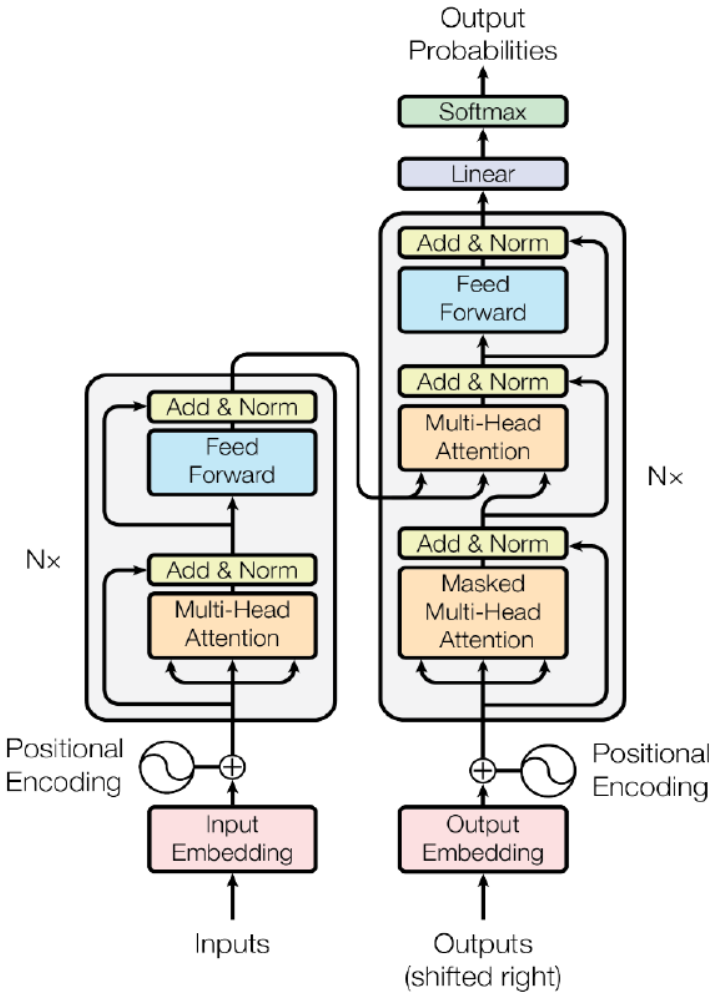

Figure 3.5.1 The Transformer model [6]

3.5.2 BERT

BERT is a pre-trained language representation model that consists of a multi-layer bidirectional Transformer encoder [19]. With BERT, Devlin et al. emphasize the importance of bidirectional pre-training for language representation models and how it improves state-of-the-art performances for many NLP tasks [19]. Implementing BERT consists of two main steps: pre-training and finetuning [19]. For the pre-training step, the BERT model is trained on a large corpus of unlabeled data over many diverse pre-training tasks, then, the model is initialized using the pre-trained parameters, and all the parameters are finetuned using labeled data from a specific task [19]. In terms of input/output representations for BERT, the first token in every sequence is a special classification token dented by $[CLS]$, and a special $[SEP]$ token is used to differentiate between input sentences [19]. Moreover, in the pre-training phase, BERT uses an MLM task that uses a fill-in-the-blanks method by which it "masks" a percentage of the input tokens and then predicts the masked tokens. Furthermore, BERT is also pre-trained on a Next Sentence Prediction (NSP) task to enhance BERT's ability to understand the relationship between two sentences [19]. Figure 3.5.2 illustrates both the pre-training and finetuning steps of the BERT model.

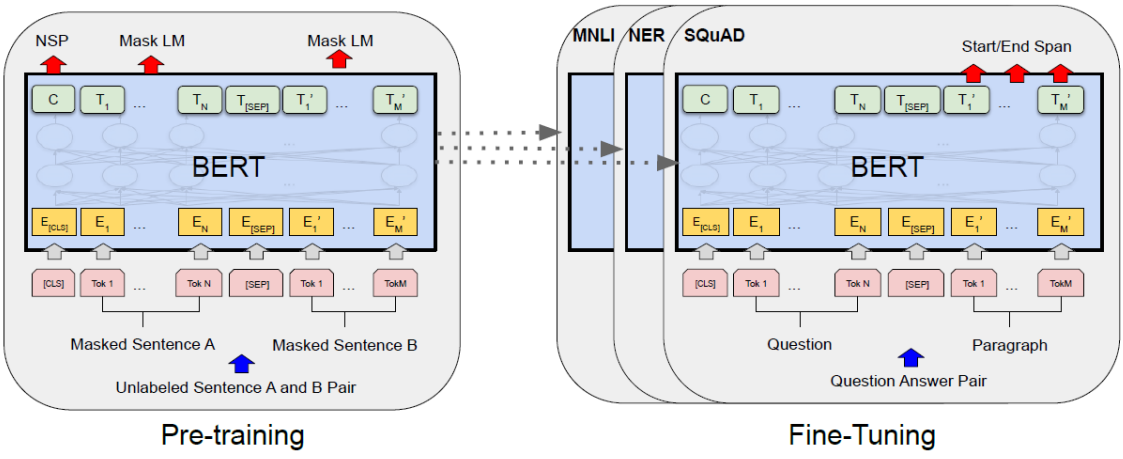

Figure 3.5.2 BERT model [19]

3.5.3 SBERT and Sentence Transformers

The emergence of BERT [19] has led to unprecedented improvements in NLP and language models. However, BERT suffers from a computational overhead, making it unsuitable for the STS task [42]. Reimers et al. [42] introduced a modified version of BERT called Sentence-BERT (SBERT) to overcome this obstacle. SBERT uses Siamese and triplet networks to extract semantically meaningful embedding from the input summaries [42]. To achieve this, Reimers et al. [42] add a pooling operation to the output of BERT-based models to produce fixed-sized sentence embeddings, which are then multiplied using cosine similarity to find the semantic similarity between the two sentences [42]. For example, consider an embedding for a reference summary u and an embedding for a candidate summary v, a pooling operation is used to produce semantically meaningful embeddings of both u and v. A cosine similarity function, $cosine\_sim(u, v)$ is applied to compute a similarity score within the range of -1 and 1 [42]. Figure 3.5.3 illustrates SBERT's architecture.

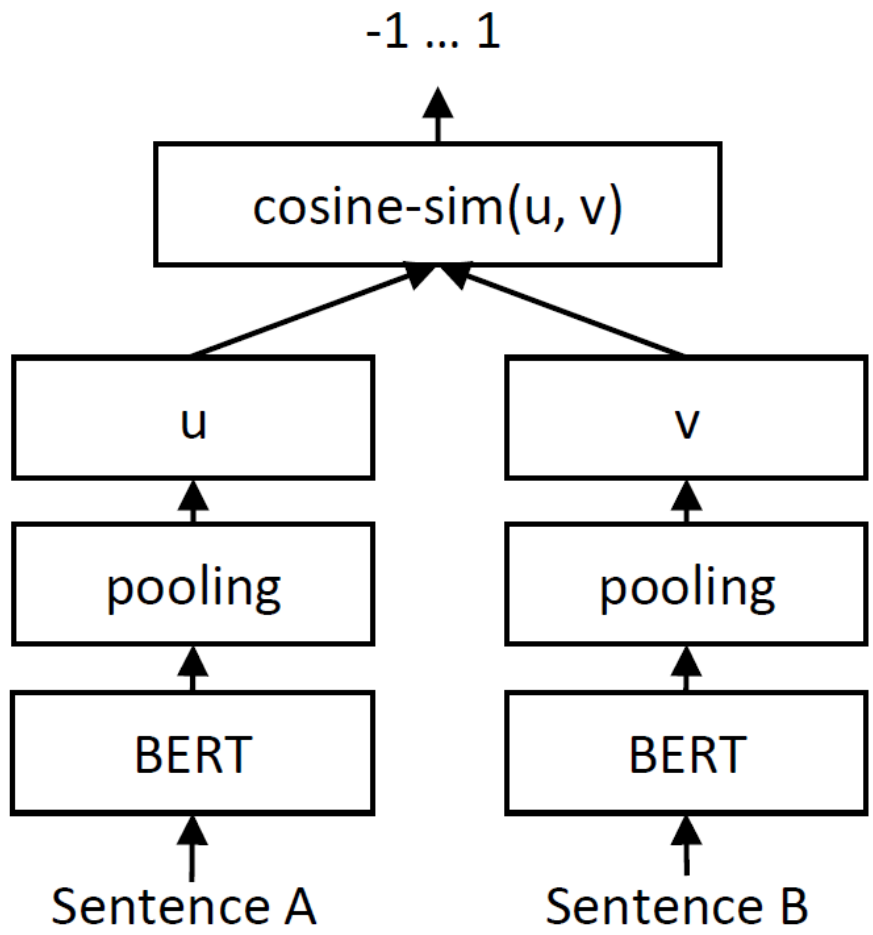

Figure 3.5.3 SBERT architecture [42]

The SBERT research also led to the emergence of Sentence-Transformer [43], a powerful Python framework that uses the SBERT [42] method to produce semantically meaningful embeddings, which are then used to determine the semantic similarity between two pieces of text.

3.5.4 RoBERTa

Models such as BERT [19] and GPT-2 [8] have brought significant gains to the field of NLP. However, their computationally expensive training makes it difficult to determine how many benefits they bring or how much these models can be trained

[44]. The creation of RoBERTa by Liu et al. [44] serves as a study that aims to experiment with different design choices and hyperparameters to create a model that can train longer with large batches over relatively more data, removes the need for next sentence prediction, can train on longer sequences and, apply the masking pattern to the training data dynamically rather than statically compared to the traditional BERT model [44]. Additionally, the authors train RoBERTa on a novel custom dataset called CC-News [44]. RoBERTa's design choices helped to achieve state-of-the-art results on several important NLP tasks, outperforming models like BERT [19] and XLNet [45].

3.5.5 DeBERTa

The DeBERTa model is a novel architecture that aims to improve upon BERT [19] and RoBERTa [44] by applying two new techniques: Disentangled attention and Enhanced mask decoder [46]. He et al. [46] implemented a disentangled attention algorithm representing each word in two vectors encoding position and content information [46]. Based on content and relative position information, the attention weights among the words are then computed by disentangled matrices [46]. The importance of relative position information resides in the strength of the dependency between two words. Moreover, as previously mentioned, BERT [19] uses MLM, where the model is asked to predict the next masked word based on its surroundings. Although DeBERTa uses content and positional information from the disentangled attention algorithm, it does not yet have the absolute positions of those words. The enhanced mask decoder technique provides the absolute positional information of the words for more accurate predictions of the masked tokens [46]. The final method applied by DeBERTa is a technique to improve the model's generalization on downstream NLP tasks by proposing a novel adversarial training method [46].

3.5.6 Improving Upon BERT and XLNet with MPNet

With BERT, Devlin et al. [19] introduced an MLM, which proved to be one of the most successful pre-training models. However, BERT suffered from neglecting dependencies between masked tokens, which caused a discrepancy between pre-training and finetuning [45]. To overcome this limitation, Yang et al. [45] proposed XLNet, a generalized autoregressive pre-training method that adopts a PLM method [45]. The XLNet PLM model is trained to predict one token given a preceding token in some random order [47]. For example, for a sentence $x = (x_1, x_2, x_3, \ldots, x_n)$ with length n, there are $n!$ possible permutations for sentence x. Using the PLM method, XLNet is able to capture bidirectional context to predict a token [46]. Furthermore, XLNet uses a two-stream self-attention mechanism that produces a content and query representation for a given sentence x_t [45]. Figure 3.5.6.1 gives an example of PLM.

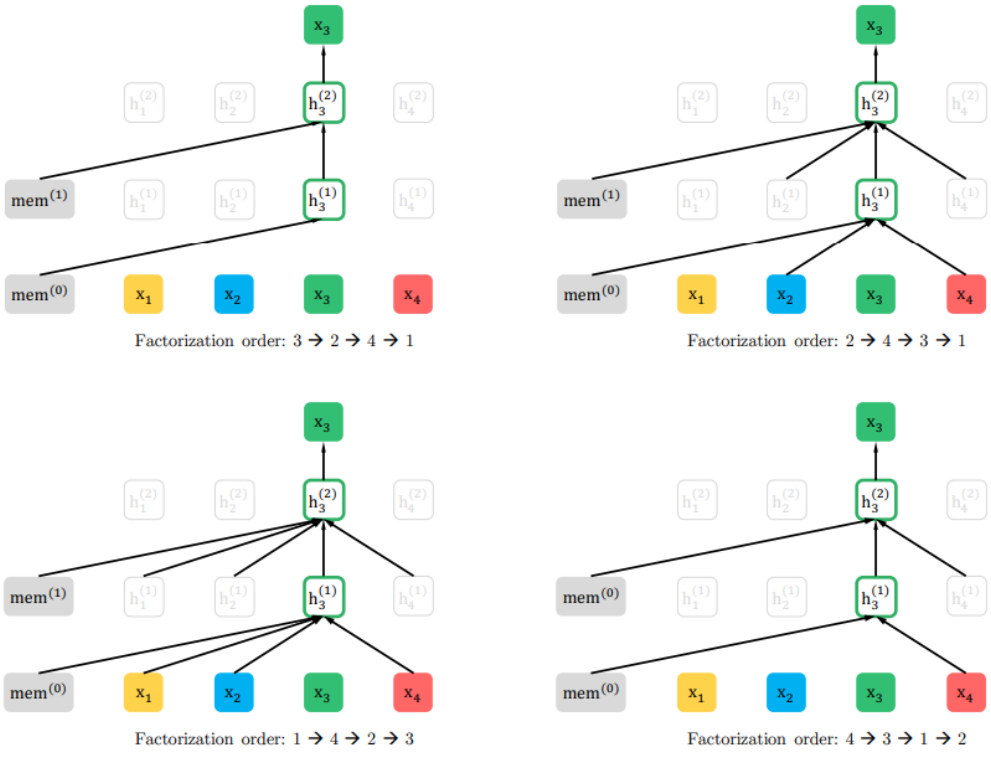

Figure 3.5.6.1 A PLM illustration to predict x_3 given an input x with different factorization orders [45]

With the power of PLM, XLNet achieved state-of-the-art results that outperformed BERT on several NLP tasks [45]. However, similar to BERT, XLNet also suffered from a drawback: it could not use the complete positional information of the sentence. In other words, each token could only see the preceding tokens in a permuted sequence, which resulted in a positional discrepancy problem [48]. To overcome BERT's dependency problem and XLNet's positional discrepancy problem, Song et al. [48] proposed MPNet, a pre-trained language model that can overcome the limitations of both BERT and XLNet [48]. MPNet can combine MLM and PLM to gain the full sentence's positional and dependency information [49]. Figure 3.5.6.2 illustrates both the structure of MPNet and the masking in two-stream self-attention.

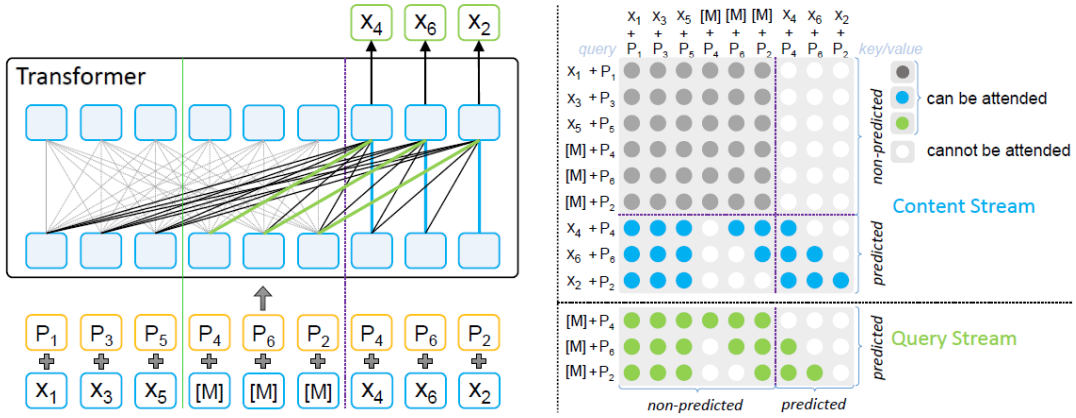

Figure 3.5.6.2 The structure of MPNet and two-stream self-attention [49]

For a given sentence $x = (x_1, x_2, x_3, x_4, x_5)$ with permutation $(x_1, x_3, x_5, x_2, x_4)$, the tokens in the right part (x_4, x_6, x_2) are selected as predicted tokens. And the non-predicted tokens are denoted as follows $(x_1, x_3, x_5, [M], [M], [M])$ with corresponding positional information as $(p1, p3, p5, p4, p6, p2)$ [49]. The produced non-predicted tokens, with the addition of positional information, allow MPNet to harness the advantages of both BERT and XLNet by gaining more information about the sentence and the tokens to be predicted [49].

MPNet is also a widely used model in the Sentence-Transformer library [43]. The previously mentioned advantages of MPNet allow the model to produce semantically meaningful embedding, which is useful in measuring the semantic similarity between a reference summary and a candidate summary.

3.5.7 T5

Transfer learning enables us to use the knowledge gained from pre-training a model on a large dataset and then fine-tuning that model for a specific task. Applying transfer learning to NLP tasks has led to many methods that achieve state-of-the-art results in multiple tasks, including abstractive text summarization. Thus, Raffel et al. [11] sought to explore the limits of applying transfer learning methods to the area of NLP, resulting in the creation of the T5 model [11]. The main goal of this model is to turn every text-processing problem into a text-to-text problem, meaning that both the input to the model and the output of the model is a text [11]. Figure 3.5.7 below provides an overview of the T5 model.

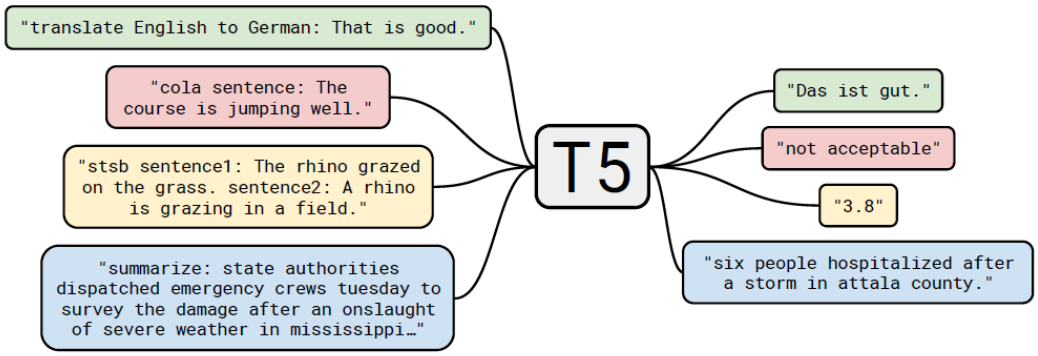

Figure 3.5.7 An overview of the T5 text-to-text approach with multiple NLP tasks. The blue-colored bubble is an example of a summarization task [11]

The T5 text-to-text model follows the encoder-decoder Transformer architecture proposed by Vaswani et al. [6]. However, there are some slight changes compared to the original Transformer model. Specifically, the authors removed the Layer Norm

bias by placing it outside the residual path and used a different positional embedding method [11]. Moreover, the encoder and the decoder of the Transformer model used by Raffel et al. [11] comprise 12 blocks where each block consists of self-attention, encoder-decoder attention, and feedforward network [11]. Furthermore, the T5 model is pre-trained on a large textual corpus named C4 (Colossal Clean Crawled Corpus). This corpus was created and cleaned by the authors, and it consists of hundreds of gigabytes of clean text data from the internet [11]. The main advantage of this text-to-text framework is its flexibility; it allows the usage of the same model with the same hyperparameters and loss function on any NLP task [11]. The T5 model comes in a wide range of sizes:

- **Base:** 220 million parameters.
- **Small:** 60 million parameters.
- **Large:** 770 million parameters.
- **3B:** 2.8 billion parameters.
- **11B:** 11 billion parameters.

The T5 model and its variants have been tested on various NLP tasks with their respective datasets. The following table shows the results of using the T5 model variants on abstractive summarization task with the CNN/DM dataset [21] by evaluating their performance with the ROUGE metric as shown in the paper by Raffel et al. [11]:

Table 3.5.1 T5 performance on summarization task

Model	**ROUGE-1**	**ROUGE-2**	**ROUGE-L**	**Dataset**
Previous best [11]	43.47	20.30	40.63	CNN/DM
T5-small [11]	41.12	19.56	38.35	CNN/DM
T5-Base [11]	42.05	20.34	39.40	CNN/DM
T5-Large [11]	42.50	20.68	39.75	CNN/DM
T5-3B [11]	42.72	21.02	39.94	CNN/DM
T5-11B [11]	43.52	21.55	40.69	CNN/DM

3.5.8 BART

Proposed by Lewis et al. [12] from Meta AI, BART is a Transformer-based pre-trained language model defined as a denoising autoencoder [12]. It is a pre-trained seq2seq model that can be finetuned on several downstream tasks such as text summarization, machine translation, and question answering [12]. BART's architecture follows an encoder-decoder model in which the encoder is a bidirectional encoder, similar to BERT [19], and a left-to-right autoregressive decoder, like GPT-2 [8]. Pre-training BART comes in two stages: a nosing function is used to corrupt the text arbitrarily, and a seq2seq model is used to reconstruct the corrupted text to its original state [12]. Figure 3.5.8 illustrates BART's architecture.

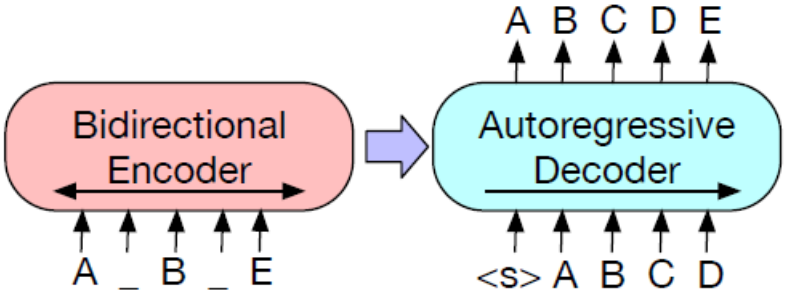

Figure 3.5.8 BART's bidirectional encoder (left) and autoregressive decoder (right) [12]

A noise function corrupts an input document by replacing some text spans with mask symbols. Then, a bidirectional model encodes the corrupted document, and an autoregressive decoder calculates the original text's likelihood [12]. Regarding text summarization, BART has achieved state-of-the-art results on two text summarization datasets, CNN/DailyMail [21] and XSUM [26].

3.5.9 SummEval Toolkit

The text summarization task has gained much momentum among NLP researchers in recent years. However, the resources needed to conduct diverse experiments in this field remain scarce [50]. To solve this problem, Fabbri et al. [50] sought to introduce a toolkit that aims to aid text summarization researchers. We used this toolkit to create a custom dataset for our work, which will be explained in further detail in Chapter 4 [50], [51].

3.6 Evaluation Metrics and Techniques

There are a wide variety of metrics that evaluate abstractive text summarization results. The aim is to measure the quality of a candidate summary generated by a model against a reference summary that humans create. In this section, we will explore automatic evaluation metrics and meta-evaluation methods.

3.6.1 ROUGE

Recall-Oriented Understudy for Gisting Evaluation, or simply ROUGE, is a popular and widely used summary evaluation metric. It is an automatic evaluation metric that measures the quality of a generated summary by comparing it to a reference summary created by a human [15]. It evaluates the summaries by counting the overlapping n-grams between the generated and reference summaries [15]. The three main types of ROUGE, namely, ROUGE-1, ROUGE-2, and ROUGE-L, are the most widely used

metrics for evaluating the quality of abstractive summarization models in numerous papers. ROUGE-N is defined as follows:

$$ROUGE\text{-}N = \frac{\sum_{S\epsilon\{ReferenceSummaries\}} \sum_{gram_n \in S} count_{match}(gram_n)}{\sum_{S\in\{ReferenceSummaries\}} \sum_{gram_n \in S} count(gram_n)} \quad (3.6.1)$$

To examine how the ROUGE metric works, let us consider the sentence "I really loved reading the Hunger Games" as a model-generated summary and "I loved reading the Hunger Games" as a human reference summary to calculate ROUGE-1, let *N = 1*, and since it is clear that ROUGE-N is a recall-oriented metric, meaning that the denominator of the equation contains the total number of n-grams in the reference summary, so the words "I," "loved," "reading," "the," "Hunger," "Games" are present in both model and reference summaries, then, in this case, ROUGE-1 = 6/6 = 1. Another example is a model-generated summary, "The cute dog is playing with a ball," and a reference summary, "A lovely pet enjoys playing with the ball." Here, the words "playing," "with," and "ball" overlap, so ROUGE-1 = 3/8 = 0.38.

3.6.2 BLEU

BLEU (Bilingual evaluation understudy) [16] is a popular automatic evaluation metric used to evaluate the quality of machine translation and text summarization models. Like the ROUGE metric [15], BLEU [16] follows an n-gram-based approach. It calculates a precision measure that counts the maximum number of overlapping words in a reference sentence and divides them by the total number of words in a candidate sentence. It also applies a brevity penalty determined by the reference and candidate sentences [16].

$$BP = \begin{cases} 1 & if\ c>r \\ e^{\left(1-\frac{r}{c}\right)} & if\ c \leq r \end{cases} \quad (3.6.2.1)$$

Then,

$$BLEU = BP \cdot \exp(\sum_{n=1}^{N} w_n \log p_n) \qquad (3.6.2.2)$$

For example, let us consider the sentence "playing video games is extremely fun" as a reference sentence and "playing video games is a waste of time" as a candidate sentence for $N = 1$ case. Since only "playing," "video", "games" and "is" words overlap between the reference and candidate sentences, the precision p_n of the n-gram is calculated using Equation (3.6.2.2). Thus 4/8 = 0.5. Moreover, the brevity penalty is calculated using Equation (3.6.2.1), which equals 1 since $c > r$. So, the final BLEU score is calculated as (0. 5*1) = 0. 5.

3.6.3 BERTScore

BERTScore is an increasingly popular automatic evaluation metric used to measure the quality of text generation models, and recently, it has gained much traction in text summarization [28]. BERTScore calculates the similarity score for each token in a candidate summary with each token in a reference summary [18]. However, compared to traditional metrics for summarization tasks, like ROUGE, which calculates the exact matches of overlapping n-grams between a candidate and a reference summary, BERTScore calculates the similarity scores using BERT's [19] contextualized embedding. Contextualized embedding forms the context of a given target word by generating different vector representations for that word in different sentences [18]. An overview of the BERTScore metric is given in Figure 3.6.3.

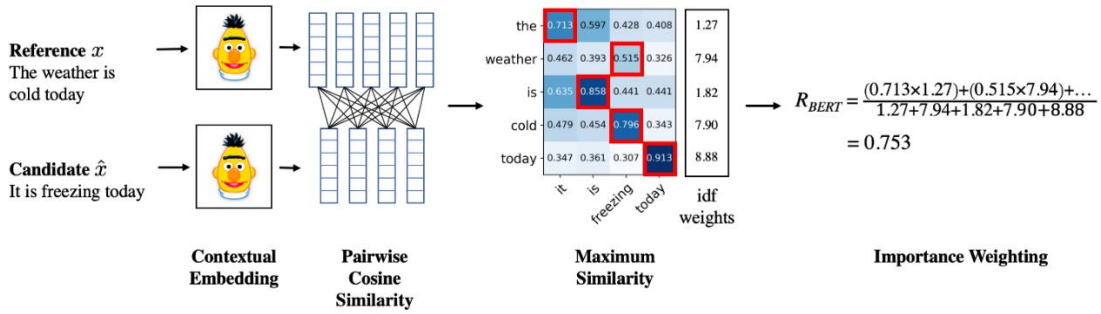

Figure 3.6.3 An Illustration of computing BERTScore Recall [18]

In terms of calculating the similarity measure between a reference token x_i and a candidate token $\hat{x}_j$, BERTScore uses pairwise cosine similarity, which is defined as $x_i^T \hat{x}_j / ||x_i|| \, ||\hat{x}_j||$, with pre-normalized vectors to reduce the computation to the inner product $x_i^T \hat{x}_j$. Furthermore, BERTScore computes recall for each token in x to each toke in $\hat{x}$ and it computes the precision of each token in $\hat{x}$ to each token in x. Finally, it combines both recall and precision to compute the F1 measure. Recall, precision, and F1 are defined as follows:

$$R_{BERT} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} x_i^T \hat{x}_j \qquad (3.6.3.1)$$

$$P_{BERT} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} x_i^T \hat{x}_j \qquad (3.6.3.2)$$

$$F_{BERT} = 2 \frac{P_{BERT} \cdot R_{BERT}}{P_{BERT} + R_{BERT}} \qquad (3.6.3.3)$$

3.6.4 MoverScore

MoverScore is an automatic evaluation metric proposed by Zhao et al. [29]. Like BERTScore [18], MoverScore utilizes contextualized embeddings obtained from a

BERT [19] model to measure the semantic distance between a model-generated candidate summary and a human-generated reference summary. To measure the semantic distance, Zhao et al. [29] use a distance formula known as Word Mover's Distance (WMD) [30], which semantically aligns similar words and finds the relationship and flow between them. For example, if we want to compute the semantic distance d between two n-gram sequences, x^n and y^n, with a transportation cost matrix C, then the aim of WMD is to minimize the transportation cost between n-gram sequences x^n and y^n with the weights of these n-grams being denoted by f_{x^n} and f_{y^n}. The WMD formula is given as follows:

$$WMD(x^n, y^n) := min_{F \in \mathbb{R}^{|x^n| * |y^n|}} \langle C, F \rangle \qquad (3.6.4)$$

$$s.t\ F1 = f_{x^n}, \qquad F^T 1 = f_{y^n}$$

Although both BERTScore [18] and MoverScore [29] use contextualized embeddings from a BERT [19] model, they differ in the way that they semantically map related sequences to each other. BERTScore [18] uses a one-to-one alignment where one word in a sequence travel to a single semantically similar word [29]. On the contrary, MoverScore [29] relies on a many-to-one alignment, where multiple words in one sequence are mapped to semantically similar words in another sequence [29]. This is accomplished by the minimization effort made by equation (3.6.4). Figure 3.6.4 gives an illustrated example of the problem.

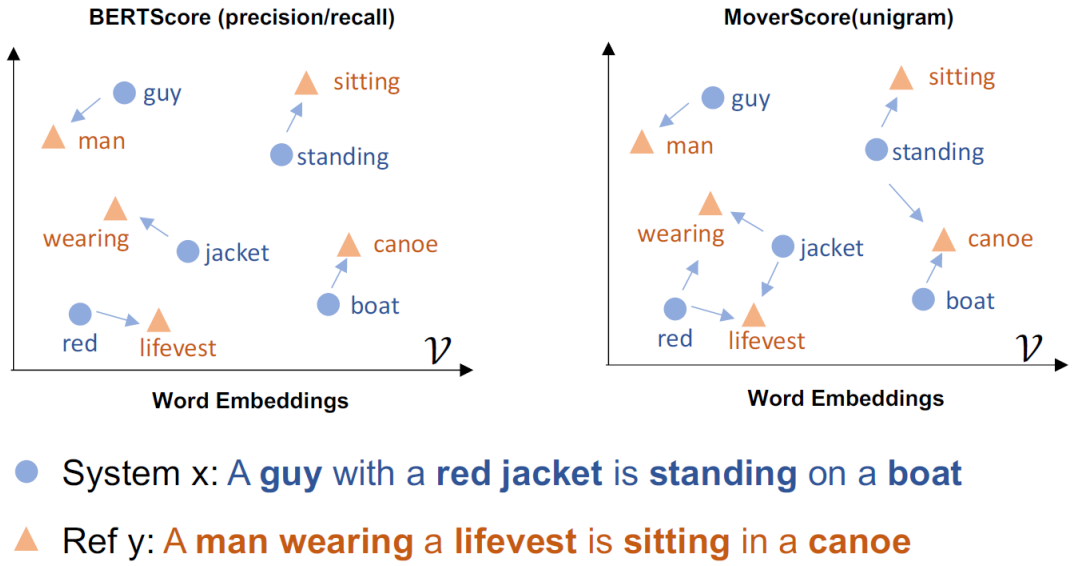

Figure 3.6.4 Difference between BERTScore and MoverScore [29]

3.6.5 NUBIA

NUBIA is a text-generation evaluation metric that adopts a purely model-based structure. In other words, it consists entirely of pre-trained language models rather than mathematical structures [31]. NUBIA consists of three Transformer models and one feedforward neural network model where the Transformer-based models extract specific features from the reference and candidate sentences. The feedforward model acts as an aggregator that assigns a score based on the given neural features [31]. Figure 3.6.5 gives an overall view of the NUBIA metric architecture.

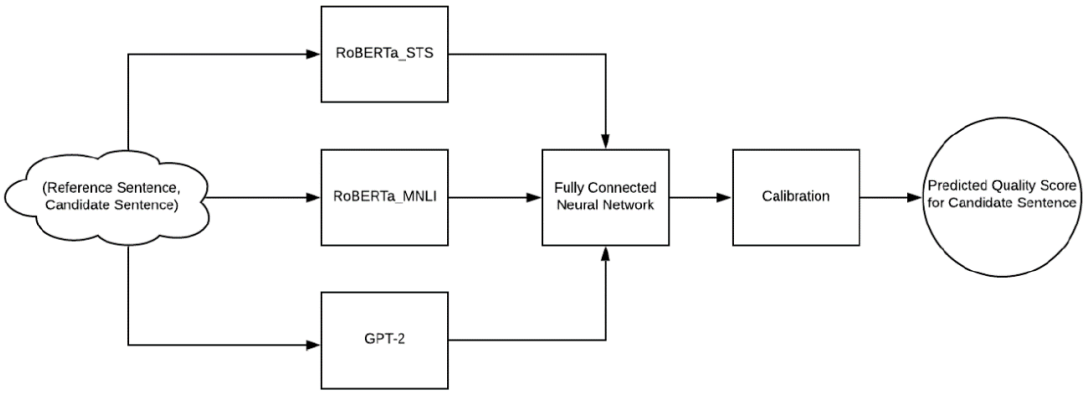

Figure 3.6.5 The NUBIA metric architecture [31]

The first two Transformer-based feature extractor models are the 'RoBERTa_STS' and the 'RoBERTa_MNLI.' These models finetune a RoBERTa [44] pre-trained model, an improved version of the BERT [19] model. The 'RoBERTa_STS' model is finetuned to extract a semantic similarity feature between a reference sentence and a candidate sentence, where it produces a similarity score between 0 and 5 by using the STS-B dataset [52]. Moreover, the 'RoBERTa_MNLI' model is trained on the MNLI natural language inference dataset [53], [54], where it measures logical inference of the sentences based on three labels: *Contradiction*, *Neutral*, and *Entailment*. Finally, the last feature extractor is a GPT-2 [8] model that measures sentence legibility, which checks the grammatical correctness of the sentences. All of these features are fed into a neural network aggregator that is trained on different versions of the WMT [55] machine translation dataset that utilizes the power of human annotation [31]. The final score is then calibrated to ensure it is between 0 and 1 [31].

3.6.6 REALSumm Meta-Evaluation

REALSumm is a meta-evaluation tool proposed by Bhandari et al. [56] to measure the reliability of automatic evaluation metrics used to evaluate summarization systems. It consists of the following:

- Summaries are generated from 14 abstractive and 11 extractive summarization models, comprising 25 summarization models [56].
- Evaluation scores from several metrics, including ROUGE [15], BERTScore [18], and MoverScore [29],[56].
- Human evaluation scores for a 100 sample from the CNN/DailyMail dataset [21],[56].

To collect human judgment scores, Bhandari et al. [56] used the LitePyramid [57] method, which is a crowdsource manual evaluation method that uses Semantic Content Units (SCUs) to measure how much information is shared between a reference summary and a candidate summary [56]. REALSumm contains a sample of 100 documents obtained from the test set of the CNN/DailyMail dataset [21], where each sample contains a reference summary and summaries generated from multiple summarization models along with their metric and human judgment scores [56]. In essence, REALSumm enables us to conduct experiments to measure the correlation between our SEAScore metric and other metrics, as well as human judgment scores, both on a summary level and a system level [56]. The summary-level and system-level correlations are calculated by equations (3.6.6.1) and (3.6.6.2), respectively:

$$K_{m_1 m_2}^{sum} = \frac{1}{n} \sum_{i=1}^{n} \left(K\left(\left[m_1(s_{i1}) \dots m_1(s_{iJ}) \right], \left[m_2(s_{i1}) \dots m_2(s_{iJ}) \right] \right) \right) \tag{3.6.6.1}$$

$$K_{m_1 m_2}^{sys} = K\left(\begin{bmatrix} \frac{1}{n} \sum_{i=1}^{n} m_1(s_{i1}) \dots \frac{1}{n} \sum_{i=1}^{n} m_1(s_{iJ}) \end{bmatrix}, \begin{bmatrix} \frac{1}{n} \sum_{i=1}^{n} m_2(s_{i1}) \dots \frac{1}{n} \sum_{i=1}^{n} m_2(s_{iJ}) \end{bmatrix} \right) \tag{3.6.6.2}$$

Where the quality of a system sys is determined by the average human score received by sys [56]. That is illustrated by equation (3.6.6.3) below:

$$HScore_{mean}^{sys_j} = \frac{1}{n}\sum_{i=1}^{n} humanScore(s_{ij}) \quad (3.6.6.3)$$

Where K is the correlation, m_i is a metric, and s_{iJ} represents the J^{th} generated summary of an i^{th} source document [56]. In terms of correlation methods, we will use the Pearson correlation, which measures a linear relationship between variables, and the Kendall rank correlation coefficient, which measures the strength of dependence between variables.

To give a more detailed explanation, the data in REALSumm consists of 100 CNN/DailyMail source documents. Each sample contains a reference summary from the dataset and a system-generated summary from the 25 summarization systems. In other words, let D be the CNN/DailyMail dataset [21], and each source document d_i in D will have a system-generated summary s_{iJ} from each of the 25 summarization systems that the REALSumm tool uses. Moreover, each sample of the 100 documents contains evaluation metric and human judgment scores. In addition, REALSumm enables us to integrate our SEAScore metric into the tool and compute a SEAScore result for each of the 100 samples and merge them with the original data. The following tables demonstrate a sample of the data for a single document, where each table uses a different summarization model to generate a summary for the same source document. Of the 25 models, let us use the outputs from a variation of T5 [11] and BART [12], the original structure is in JSON format, but for simplicity, we will use a regular table:

Table 3.6.1 Example 1 of the REALSumm Evaluation

Doc_id	0
Ref_summ	Anuradha Koirala and 425 young women and girls have been sleeping outdoors because of aftershocks. Pushpa Basnet and 45 children she cares for were forced to evacuate their residence . Seven other CNN Heroes and their organizations now assisting in relief efforts.
t5_out_large summary	cnn heroes anuradha koirala and pushpa basu are helping earthquake survivors . koirala's hiv/aids hospice wall has crumbled.
rouge_1_f_score	0.2
rouge_2_f_score	0.10345
rouge_l_f_score	0.2
bert_f_score	0.2265099585056305
mover_score	0.2040342292517625
litepyramid_recall	0.2
seascore	0.6448841458477546

Table 3.6.2 Example 2 of the REALSumm Evaluation

Doc_id	0
Ref_summ	Anuradha Koirala and 425 young women and girls have been sleeping outdoors because of aftershocks. Pushpa Basnet and 45 children she cares for were forced to evacuate their residence . Seven other CNN Heroes and their organizations now assisting in relief efforts.
bart_out summary	Two CNN Heroes are among the earthquake survivors in Kathmandu, Nepal. Anuradha Koirala , who rescues victims of sex trafficking , has a rehabilitation center that is home to 425 young women and girls . Pushpa Basnet and the 45 children she cares for were also forced to evacuate their residence. Several CNN Heroes have been assisting in relief efforts in Nepal .
rouge_1_f_score	0.6
rouge_2_f_score	0.42857
rouge_l_f_score	0.58
bert_f_score	0.5088735
mover_score	0.2040342292517625
litepyramid_recall	0.6
seascore	0.7134444588737097

From Tables 3.6.1 and 3.6.2, the **Doc_id** is the id number of the document, **Ref_summ** is the reference summary from the dataset, and **t5_out_large** and **bart_out** are the

model-generated summaries of the 0^{th} document. The rest are evaluation scores for each metric, including ROUGE [15], BERTScore [18], and MoverScore [29]. There are other metrics in REALSumm, but we filtered out some metrics. Finally, the **litepyramid_recall** is the human judgment score for the model-generated summaries of the 0^{th} document.

For our work, we will only consider the outputs of abstractive summarization models from REALSumm, including T5 [11], BART [12], Pointer-Generator Network [5], presumm [27], fastabsrl [22], bottom-up [23], unilm [58], two-stage-rl [59], and semsim [60]. Moreover, we will only consider ROUGE [15], BERTScore [18], MoverScore [29], and litepyramid [57] as our metrics of choice.

3.7 Datasets

In this section, several important datasets will be presented along with their relative tasks.

3.7.1 Text Summarization Datasets

The general structure of text summarization datasets involves a source document that is the text to be summarized and a reference summary that humans usually make to that particular document. Several popular summarization datasets cover various domains, including news articles, Wikihow articles, medical articles, and scientific research articles. One of the most popular summarization datasets is the CNN/DailyMail [21] dataset which consists of approximately 300K news articles written by journalists from CNN and the DailyMail. This dataset supports both extractive and abstractive summarizations.

3.7.2 MNLI Dataset

MNLI, or Multi-Genre Natural Language Inference dataset, is a corpus that is developed to evaluate sentence understanding by machine learning models [54]. This dataset is more diverse than traditional NLI datasets as it uses sentences from a wide range of domains to capture the complexity of modern English [54]. The MNLI dataset consists of 433K examples, and it labels two sentences with one of the following labels: *Entailment, Neutral, or Contradiction* [54]. Thus, this corpus helps in recognizing textual entailment between sentences.

3.7.3 CoLA Dataset

The CoLA corpus evaluates the grammatical acceptability of a sentence that is produced by artificial neural networks [61]. It consists of a collection of 10K sentences with acceptability labels, which are two labels, either linguistically acceptable or linguistically unacceptable [61]. Figure 3.7.3 demonstrates some examples of how CoLA determines linguistic acceptability.

Label	Sentence
*	The more books I ask to whom he will give, the more he reads.
✓	I said that my father, he was tight as a hoot-owl.
✓	The jeweller inscribed the ring with the name.
*	many evidence was provided.
✓	They can sing.
✓	The men would have been all working.
*	Who do you think that will question Seamus first?
*	Usually, any lion is majestic.
✓	The gardener planted roses in the garden.
✓	I wrote Blair a letter, but I tore it up before I sent it.

Figure 3.7.3 CoLA dataset example, each sentence is labeled as either (* = not acceptable) or (✓ = acceptable) [61]

CHAPTER 4

METHODOLOGY

In this chapter, we present the implementation details of our evaluation metric and the training implementation of the summarization model.

4.1 SEAScore: Similarity, Entailment, and Acceptability Score

SEAScore is our newly proposed evaluation metric that measures the quality of summaries that abstractive text summarization models generate. It takes a candidate summary generated by a summarization model and a reference summary created by humans and outputs an evaluation score between 0 and 1. Kane et al. [31], who developed the NUBIA metric, used a similar approach. SEAScore adopts a fully model-based architecture that consists of three LLMs and a linear regression aggregator model. The LLMs produce a set of features that inspect the quality of a candidate summary against a reference summary. These features are then fed to a linear regression model trained on our custom dataset. The linear regression model acts as an aggregator that produces a weighted sum result based on the features LLMs extract. Each LLM accepts a candidate summary and a reference summary and produces a score reflecting the quality of the features: semantic textual similarity (STS), natural language inference (MLNI), and linguistic acceptability (CoLA). The result is a total of eight numerical features and a final score that represents the score given by the aggregator. Figure 4.1 provides an overview of the SEAScore metric.

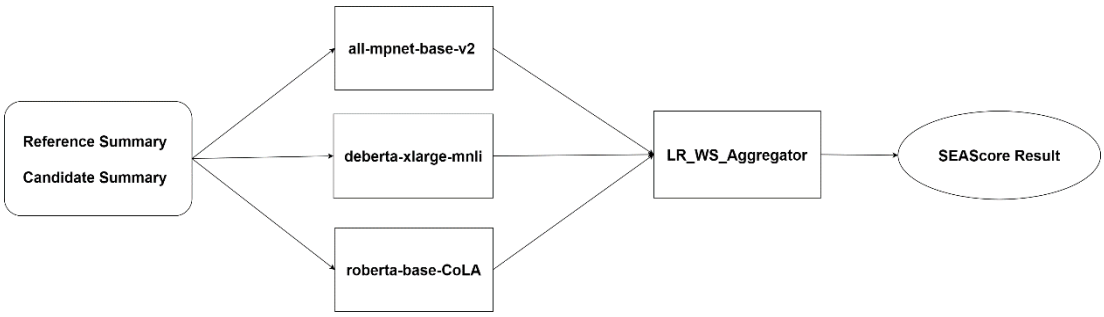

Figure 4.1 SEAScore metric architecture

In the metric, we use an MPNet-based sentence transformer to measure semantic similarity [62], a DeBERTa-based MNLI model to measure the degree of entailment [63], and a RoBERTa-based CoLA model to measure linguistic acceptability [64]. These models were obtained using the HuggingFace Transformer liberty [65]. Moreover, the Transformer models produce a total of eight features. The following sections will investigate how SEAScore conducts the evaluation process using an example candidate and reference summary. The same example will be used for each LLM, and the aggregator will give the final result.

4.1.1 Semantic Similarity with MPNet

The first and most important feature of the SEAScore metric is a semantic similarity score between a candidate summary and a reference summary. To achieve this, we used an MPNet-based [48] sentence transformer [43] called *all-mpnet-base-v2* [62]. This sentence transformer, which adopts an SBERT architecture [42], uses an MPNet [48] model and finetunes it on a large and diverse corpus that consists of approximately 1.1 billion sentence pairs [62]. The result is a sentence transformer model that can produce semantically meaningful embeddings for a given summary. The model takes two summary pairs, a candidate and a reference summary, and creates embeddings for the two summaries. Then, a cosine similarity score is computed using the embeddings obtained by the MPNet sentence transformer model, resulting in a semantic similarity score that illustrates the similarity between the two summaries.

To take a closer look at how *all-mpnet-base-v2* [62] measures semantic similarity, we will go through an example that illustrates the process behind producing a similarity score. Let $S1$ be a candidate summary and $S2$ be a reference summary, the model *all-mpnet-base-v2* will produce an embedding matrix for both $S1$ and $S2$ respectively, let us call them $S1_{emd}$ and $S1_{emd}$. Finally, a cosine similarity function, $cosine\_sim(S1_{emd}, S2_{emd})$, will compute a semantic similarity score between the range 0 and 1, which indicates the degree of similarity between $S1$ and $S2$. Figure 4.1.1 and Table 4.1.1 below give a simple summary of the process with an example.

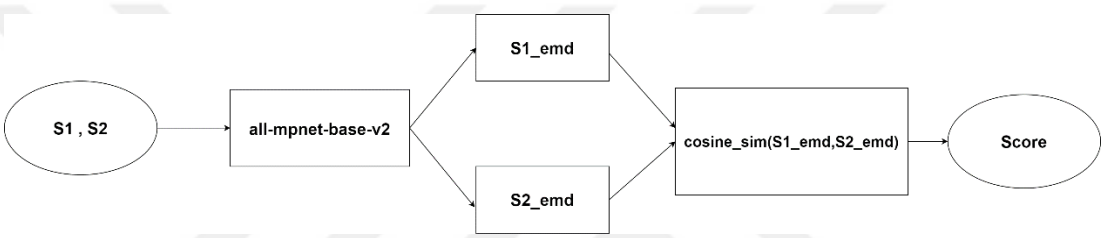

Figure 4.1.1 How *all-mpnet-base-v2* model computes semantic similarity

Table 4.1.1 A similarity score computed by *all-mpnet-base-v2*

Reference summary	frankie dettori to focus on the british flat racing season this summer. gregory benoist chosen to ride in france by sheik joaan al thani. ap mccoy will ride his last scottish grand national on benovillo on friday.
Candidate summary	frankie dettori will be left to focus on the british flat season this summer. boss sheik joaan al thani signed up gregory benoist to ride the horses which race under his al shaqab banner in france . the only exception will be sheik joaan's dual arc winner treve, who will continue to be partnered by veteran thierry jarnet .
Semantic similarity	80.5

4.1.2 Multi-Genre Natural Language Inference with DeBERTa

For the MNLI task, we used the *deberta-xlarge-mnli* model [63], which has 750 million parameters. This model is finetuned on the MNLI dataset [53],[54], which performs a classification task on a pair of summaries using three labels: *contradiction*, *neutral*, and *entailment*. The model takes a pair of candidate and reference summaries, uses its tokenizer to tokenize the summary pairs so that they are in the correct format, and outputs three numerical results that measure to which degree the pair of summaries belong to either of the three labels. We can use these features to gain a broader insight into the relationships between the summary pairs as to whether they contradict or entail each other. Table 4.1.2 shows how *deberta-xlarge-mnli* [63] applies the MNLI [54] classification task on the same pair of summaries from Table 4.1.1.

Table 4.1.2 MNLI classification task by *deberta-xlarge-mnli*

Reference summary	frankie dettori to focus on the british flat racing season this summer. gregory benoist chosen to ride in france by sheik joaan al thani . ap mccoy will ride his last scottish grand national on benovillo on friday.
Candidate summary	frankie dettori will be left to focus on the british flat season this summer. boss sheik joaan al thani signed up gregory benoist to ride the horses which race under his al shaqab banner in france. the only exception will be sheik joaan's dual arc winner treve, who will continue to be partnered by veteran thierry jarnet .
Contradiction score	0.68
Neutral score	12.54
Entailment score	86.78

4.1.3 Linguistic Acceptability with RoBERTa

To measure the grammatical and linguistic acceptability of a set of summary pairs, we use the *roberta-base-CoLA* model [64], which is a RoBERTa model [44] that is finetuned on the CoLA dataset [61]. The model classifies a single sentence into either linguistically acceptable or unacceptable. It outputs a percentage score for each of the two labels to better represent the degree of acceptability for the given sentence. We use this model to individually obtain the label scores for each candidate summary and reference summary. The result is four numerical scores that measure the acceptability/unacceptability of those summaries. Table 4.1.3 shows the results of

applying this classification task on the candidate and reference summaries used in Tables 4.1.1 and 4.1.2.

Table 4.1.3 CoLA classification task by *roberta-base-CoLA*

Reference summary	frankie dettori to focus on the british flat racing season this summer. gregory benoist chosen to ride in france by sheik joaan al thani. ap mccoy will ride his last scottish grand national on benovillo on friday.
Candidate summary	frankie dettori will be left to focus on the british flat season this summer. boss sheik joaan al thani signed up gregory benoist to ride the horses which race under his al shaqab banner in france. the only exception will be sheik joaan's dual arc winner treve, who will continue to be partnered by veteran thierry jarnet.
Reference unacceptability	71.39
Reference acceptability	28.61
Candidate unacceptability	33.42
Candidate acceptability	66.57

4.1.4 Aggregator

The aggregator component in SEAScore aims to receive a total of eight numeric features from the LLMs [62], [63], [64] and output a score that reflects the overall evaluation of a pair of summaries. In the following sections, we will explain the creation of a custom dataset that we used to train the aggregator and the model type of

our choice. Finally, we will use the features we obtained for the example in sections 4.1.1, 4.1.2, and 4.1.3 and produce a SEAScore result for that specific example.

4.1.4.1 Creating the Custom Dataset for Aggregator

We used the SummEval toolkit [51] to obtain 151 model-generated summaries from multiple text summarization models, including T5 [11], BART [12], PEGASUS [13], pointer-generator [5], BERTSUMABS [27]. All the models mentioned use the CNN/DailyMail dataset [21]. We manually annotated the summary pairs by giving each model feature a certain weight, such that the sum of the given custom weights equals 1. We obtained a final metric score result by using the following formula:

$$Y = f_1 w_1 + f_2 w_2 + f_3 w_3 + \cdots + f_n w_n \quad (4.1)$$

Where Y is the SEAScore metric score, f_n is the feature produced by the models that are used in SEAScore, and w_n is the custom weight given to the f_n feature. We carried out this process by reading each summary pair individually and giving them weights based on answering the following questions:

- Does the candidate summary give the same information as the reference summary?
- Is any new information presented in the model-generated summary that does not exist in the reference summary? Does it drastically change the core information of the candidate summary?
- From a human perspective, are the two summaries readable? Do they suffer from linguistic anomalies?

The final Y score represents the SEAScore metric result that considers the answers to all the questions above. Tables 4.1.4.1 and 4.1.4.2 below illustrate a small sample from

the custom dataset. For simplicity, the columns of the dataset will be named the following:

- *ss* stands for the *semantic similarity* score obtained from *all-mpnet-base-v2* [62].
- *cont*, *ntrl*, and *entl*, stand for the MNLI [44] labels *contradiction*, *neutral*, and *entailment* respectively. These features are obtained from *deberta-xlarge-mnli* [63].
- *r0*, *r1*, c0, c1 stand for *reference unacceptability*, *reference acceptability*, *candidate unacceptability*, and *candidate acceptability* respectively. These features are obtained from *roberta-base-CoLA* [64].
- *seascore* stands for the SEAScore result.

Table 4.1.4.1 A sample of the custom dataset

ss	**cont**	**ntrl**	**entl**	**r0**	**r1**	**c0**	**c1**	**seascore**
0.600952	0.56399	0.23766	0.19834	0.36095	0.63904	0.0903	0.9096	**0.4875**
0.802432	0.00114	0.98750	0.01134	0.33014	0.66985	0.34201	0.6579	**0.6729**
0.906649	0.00173	0.91036	0.08789	0.04222	0.95777	0.0758	0.9241	**0.8438**
0.915365	0.00021	0.99209	0.00769	0.54783	0.45216	0.4822	0.5177	**0.7515**
0.470855	0.09843	0.89991	0.00165	0.15630	0.84369	0.2125	0.7874	**0.4781**

Table 4.1.4.2 A sample of the weights used for each feature in Table 4.1.4.1

W1	**W2**	**W3**	**W4**	**W5**	**W6**	**W7**	**W8**
0.40	0.010	0.090	0.09	0.1050	0.1000	0.1050	0.10
0.40	0.010	0.090	0.01	0.0950	0.1500	0.0950	0.15
0.50	0.010	0.010	0.01	0.0350	0.2000	0.0350	0.20
0.50	0.010	0.095	0.01	0.1500	0.0425	0.0425	0.15
0.40	0.010	0.010	0.09	0.0950	0.1500	0.0950	0.15

For Tables 4.1.4.1 and 4.1.4.2, each row in table 4.1.4.1 corresponds to each row in table 4.1.4.2 where each feature f_n is multiplied by each weight w_n to produce a SEAScore Y.

Some important observations to consider for the custom dataset, the semantic similarity feature is the most dominant, so it largely dictates what the SEAScore result will be. Moreover, the results in the *seascore* column in Table 4.1.4.1 reflect the evaluation given by us, the human annotators, so each score represents how the human annotator evaluated each summary pair. The task of the aggregator is to automate this process such that whenever the aggregator receives a set of eight features from the LLMs [62], [63], [64], it should return a score that mimics the behavior of a human evaluator.

4.1.4.2 Training the Aggregator Model

The aggregator model in SEAScore is a regression model that takes eight features and produces an evaluation score. We trained and tested two types of regression models. Firstly, we trained a neural network model on the custom dataset. The training results for this model were not satisfactory, as it showed signs of overfitting when tested on multiple unseen examples. Another reason the neural network model was not a suitable choice for this task is that the custom dataset had a small number of examples, 151 to be exact, which can cause the model to overfit.

Instead of a neural network model, we train a simple linear regression model obtained from the scikit-learn library [66] on the custom dataset. We split the dataset using an 80-20 train-test split, obtaining a train set of 121 samples and a test dataset of 31 samples. Finally, we trained the model on the training dataset. The R^2 score on the test dataset gave 92.61% accuracy.

Using all the features obtained from sections 4.1.1, 4.1.2, and 4.1.3, Table 4.1.4.3 demonstrates the SEAScore result produced by the linear regression aggregator.

Table 4.1.4.3 SEAScore result obtained using all features and the aggregator

Reference summary	frankie dettori to focus on the british flat racing season this summer. gregory benoist chosen to ride in france by sheik joaan al thani. ap mccoy will ride his last scottish grand national on benovillo on friday.
Candidate summary	frankie dettori will be left to focus on the british flat season this summer. boss sheik joaan al thani signed up gregory benoist to ride the horses which race under his al shaqab banner in france. the only exception will be sheik joaan's dual arc winner treve, who will continue to be partnered by veteran thierry jarnet .
Semantic similarity	80.5
Contradiction score	0.68
Neutral score	12.54
Entailment score	86.78
Reference unacceptability	71.39
Reference acceptability	28.61
Candidate unacceptability	33.42
Candidate acceptability	66.57
Aggregator SEAScore result	**68.79**

The SEAScore result of 68.79 for the example in Table 4.1.4.3 indicates that the summary pairs are semantically similar and have an impressive degree of entailment, meaning they share similar core information. However, they lack linguistic acceptability as both summaries are difficult to read and understand from a human perspective.

4.2 Training Summarization Models

For our summarization models, we trained three different models, we finetuned two pre-trained models, T5 [11] and BART [12], and we trained a Transformer model [6] from scratch. We used several libraries to achieve our task, including HuggingFace (HF) Transformer library [65], TensorFlow (TF) [67], and PyTorch (PT) [68].

4.2.1 Dataset

We trained our models using the CNN/DailyMail [21] summarization dataset containing 311,971 samples. The data split for this dataset is as follows: 287,113 train set (92.0 %), 13,368 validation set (4.3 %), and 11,490 test set (3.7 %) [21]. Moreover, the CNN/DailyMail dataset consists of three columns: the *article* column that contains the complete source articles to be summarized, the *highlights* column that contains the human-created summaries in the form of bullet points, where each sentence is separated with a '*n*' end-of-sentence character, and an *id* column that holds the *id* of each article and highlight. In terms of the length distribution of the articles and highlights, Figure 4.2.1 below demonstrates the distribution of the input and target lengths of the dataset.

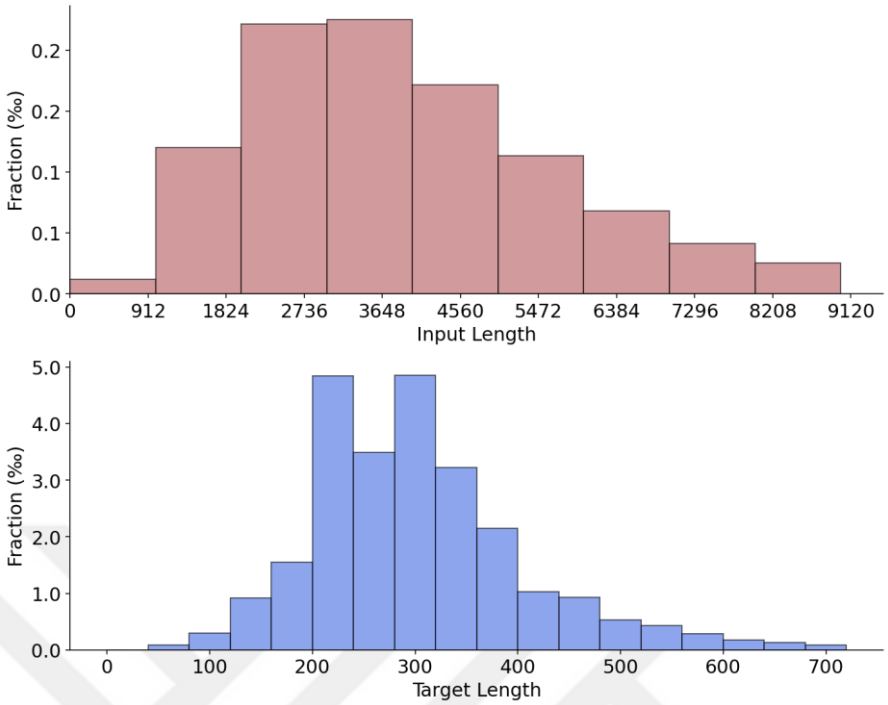

Figure 4.2.1 Input and target lengths distribution for CNN/DailyMail dataset

4.2.2 Dataset Pre-processing and Model Implementations

This section will give the implementation details of the summarization models and the CNN/DailyMail [21] dataset pre-processing for each model.

4.2.2.1 Data Pre-processing for Transformer from Scratch

We used pre-processing techniques to prepare the CNN/DailyMail dataset for the Transformer model to ensure the data was in the correct format. The data pre-processing steps included the following:

- Removing the *id* columns from the dataset since it is not needed.
- Adding [start] and [end] tokens to each highlight in the training set and the validation set to indicate the start and end of the highlights.

- Limiting the input sequence lengths of the source documents to length 2048 to reduce the input size to simplify the training process.
- Using the TF API vectorization layer to tokenize the articles and highlights of the dataset so that the inputs and the targets are in a model-readable state.

4.2.2.2 Transformer Model from Scratch Implementation

Our first summarization model is a Transformer model from scratch; we named this model the *transformer_summ*. We implemented the Transformer architecture proposed by Vaswani et al. [6], which adopted an encoder-decoder architecture that used a positional encoding method, a Multi-head Attention mechanism, and a feedforward neural network model [6]. The details of the Transformer model are explained in section 3.5.1 of Chapter 3. The difference between the model implemented in the paper "Attention Is All You Need" [6] and our *transformer_summ* model is that we used a positional embedding method instead of a positional encoding method. This method is used by François Chollet [41], and it suggests using an embedding layer that adds the positional embedding information to the corresponding input word embeddings. We added a TF embedding layer to our positional embedding method that randomly initializes its weights. Then, during training, the weights are updated as the model keeps learning, allowing the model to learn a new embedding space [41]. This method is more straightforward since it depends on learning the weights of the embedding layer.
The *transformer_summ* model is trained on the CNN/DailyMail dataset [21], where it takes an input source article and outputs a target summary. All implementations of this model were achieved using the TF API [67].

For the *transformer_summ* model, we used the Adam optimizer [69], which is an optimizer that has three parameters: the *alpha* α, which is the learning rate or can also be called the step size, the β_1, which is the exponential decay of the first moment, and the β_2, which is the exponential decay rate of the second moment [69]. Moreover, we also used a Dropout rate of 0.5, a regularization method that randomly drops units from

the neural network to prevent the model from overfitting [70]. The following Table 4.2.2.2 summarizer all the hyper-parameters for the *transformer_summ* model.

Table 4.2.2.2 Hyper-parameters for *transformer_summ* model

Hyper-parameters	**Value**
sequence_length	1024
batch_size	4
epochs	50
buffer_size	10000
Optimizer	Adam
Dropout	0.5
embed_dim	128
dense_dim	512
num_heads	4
source_vocab_size	155707
target_vocab_size	67774

4.2.2.3 Data Pre-processing for T5

To feed our data to the T5 LLM, we first need the dataset to be in an acceptable format for the T5 model. To achieve this, we used an HF library method [65] that enabled us to download a T5-based tokenizer that transforms the dataset to the required format. We limited the input (source article) and target (reference summary) sequence lengths to 1024 and 128, respectively. Moreover, the HF library abstracts the tokenization process, so we had to apply the tokenizer to the entire dataset and use it to train the model.

4.2.2.4 T5 Implementation

The T5 model, initially proposed by Raffel et al. [11], is an LLM pre-trained on an extensive data corpus. The details of the model are explained in section 3.5.7 of Chapter 3. The knowledge gained from the pretraining process allows us to finetune this model on any downstream task. We finetuned the small version of the T5 model, which we called the *t5_summ*, on the abstractive summarization task using the CNN/DailyMail dataset [21]. The small version of the T5 model contains 60 million parameters. We first obtained this model from the HuggingFace API [65], simplifying downloading LLMs using their custom HF library methods. Furthermore, we downloaded the TensorFlow (TF) version of the model, and we transformed the dataset that we obtained from section 4.2.2.3 to TF datasets and fed it into our model for the training process. Finally, we used an HF callback method that computes the ROUGE [15] scores after each training epoch.

In terms of optimizers, we used the AdamW [71] optimizer, which is a modified version of the Adam optimizer [69] that uses the implementation of L_2 regularization method; this prevents the model from overfitting by keeping the weights of the model small, and thus it helps to avoid the exploding gradient problem, which can negatively affect the performance of the model. We also used weight decay [72] to prevent overfitting. Table 4.2.2.4 summarizes the hyper-parameters used to train the *t5_summ* model.

Table 4.2.2.4 Hyper-parameters for the *t5_summ* model

Hyper-parameters	**Value**
max_input_length	1024
max_target_length	128
batch_size	2
learning_rate	2e-5
weight_decay	0.01
num_train_epochs	1
Optimizer	AdamW
Dropout	0.1
d_ff	2048
d_model	512
num_decoder_layers	6
num_heads	8
num_layers	6

4.2.2.5 Data Pre-processing for BART

To transform the CNN/DailyMail dataset [21] into a suitable form for BART, we used the same approach from section 4.2.2.3. We downloaded the BART tokenizer from the HuggingFace API [65] that tokenizes the dataset. We limited the input (source article) and target (summary) sequence lengths to 1024 and 128, respectively, and applied the BART tokenizer to each example in the entire dataset.

4.2.2.6 BART Implementation

BART is an LLM that uses an encoder-decoder architecture that consists of a bidirectional encoder and a left-to-right autoregressive decoder [12]. All details of BART are explained in section 3.5.8 of Chapter 3. Similar to the T5 model [11], BART is also a pre-trained model that can be finetuned for various text generation tasks. We finetuned a base version of the BART model, which contains 140 million parameters [12], on the abstractive text summarization task using the CNN/DailyMail dataset [21].

We called this model the *bart_summ* model. We obtained the base version of BART by downloading it from the HuggingFace (HF) API [65], and we used an HF callback method that computes the ROUGE [15] scores after each training epoch. We used the PyTorch (PT) [68] version of BART due to some practical implementation abstractions that HF provides. As our optimizer of choice, we used the AdamW [71] and added a dropout rate [70] and used weight decay [72] to prevent the model from overfitting and giving unsatisfactory results. Table 4.2.2.6 summarizes the hyper-parameters for the *bart_summ* model.

Table 4.2.2.6 Hyper-parameters for *bart_summ* model

Hyper-parameter	**Value**
max_input_length	1024
max_target_length	128
batch_size	2
learning_rate	2e-5
weight_decay	0.01
num_train_epochs	1
Optimizer	AdamW
d_model	768
decoder_attention_heads	12
decoder_ffn_dim	3072
decoder_layers	6
Dropout	0.1
encoder_attention_heads	12
encoder_ffn_dim	3072
encoder_layers	6

4.3 Hardware

All coding implementations in this study were conducted using the Google Colab environment. Google Colab provides several options for computational resources such as RAM amount, GPU, and TPU resources. We used Google Colab Pro, a paid tier that costs 163.84₺/month. This paid tier provides more powerful GPUs and increased RAM. We used a premium GPU option that significantly decreased training for our models.

CHAPTER 5

RESULTS AND DISCUSSIONS

This chapter will share the training results for the summarization models and several experiments that compare our SEAScore metric with other standard metrics.

5.1 Training Results for Summarization Models

This section will present the training results for the summarization models. We trained a Transformer model from scratch [6], the *transformer_summ* model, and we used transfer learning [11] to finetune two LLMs, *t5_summ* and *bart_summ*, on the summarization task.

5.1.1 Training Results for transformer_summ

We implemented the *transformer_summ* model by adopting the Transformer model [6] architecture illustrated in Figure 3.5.1 of Chapter 3 with the addition of a positional embedding mechanism as explained in detail in section 4.2.2.2 of Chapter 4. Moreover, we initialized the hyper-parameters for *transformer_summ* as summarized in Table 4.2.2.2. The *transformer_summ* is trained on the CNN/DailyMail [21] train set using 39625 training examples, where the model takes a source document as input and a summary of that document as a target output. We trained *transformer_summ* for 50 epochs, which took approximately 42 hours in the Google Colab environment. The training results displayed a steady decrease in the training loss values as *transformer_summ* continued training, indicating that the model was learning effectively. Figure 5.1.1 displays the training results for *transformer_summ*.

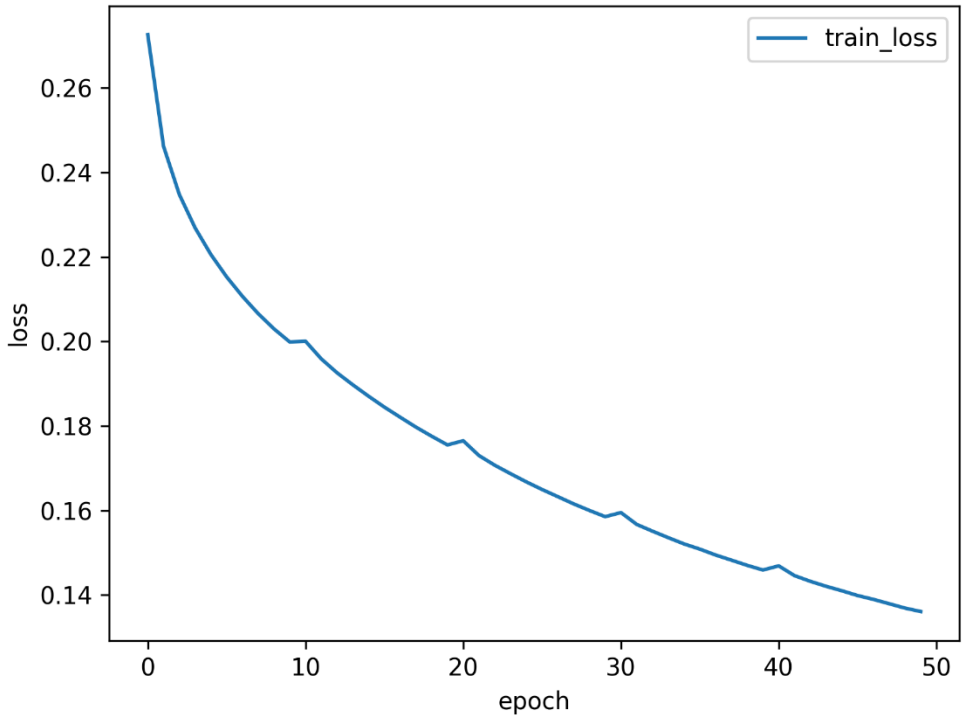

Figure 5.1.1 Training result for *transformer_summ*

5.1.2 Training Results for t5_summ and bart_summ

We finetuned T5 [11] and BART [12] LLMs on the abstractive summarization task, which we named *t5_summ* and *bart_summ*, respectively. We first obtained these models using the HF API [65] and finetuned them on the CNN/DailyMail dataset [21]. Implementation details and the summary of the hyper-parameters for each model are explained in detail in sections 4.2.2.4 and 4.2.2.6 of Chapter 4. The training process for both *t5_summ* and *bart_summ* was the same since we utilized the methods provided by the HF API [65]. We used a training set, a validation set, and a generation set for the training process. The generation set uses the validation set to enable the models to generate predictions with the help of an HF callback method and to compute a ROUGE [15] score after each training epoch; this helps in understanding how well the models perform during training. Furthermore, *t5_summ* was trained for one epoch

which took approximately 10 hours using a standard GPU from the Google Colab environment. On the other hand, *bart_summ* trained for one epoch which took four hours to complete on a premium GPU provided by Google Colab. The training results for both models are provided in Table 5.1, obtained from HF API.

Table 5.1 Training result for *t5_summ* and *bart_summ*

Training Data	**t5_summ**	**bart_summ**	**Dataset**
Training Loss	1.1086	0.961	CNN/DailyMail
Validation Loss	1.0703	0.9441	CNN/DailyMail
Epoch	1	1	CNN/DailyMail
Step	143556	143557	CNN/DailyMail
ROUGE-1	24.2706	24.5981	CNN/DailyMail
ROUGE-2	11.6659	12.307	CNN/DailyMail
ROUGE-L	20.0926	20.4524	CNN/DailyMail
ROUGE-L-SUM	20.1515	20.5108	CNN/DailyMail
Gen Len	18.9998	19.9993	CNN/DailyMail

The results for *bart_summ* have a slight edge over *t5_summ* in terms of loss values and ROUGE scores.

5.2 Metric Results for Summarization Models

This section examines results from several automatic evaluation metrics for our summarization models. We use *transformer_summ*, *t5_summ*, and *bart_summ* to generate summaries for 1000 sample source documents from the CNN/DailyMail test set, and we use the reference summaries for each source document provided in the dataset. As for metrics, we will use ROUGE [15], BLEU [16], BERTScore [18], MoverScore [29], NUBIA [31], and our SEAScore metric. For simplicity, we will only consider the F1 scores for ROUGE and BERTScore results. This experiment examines

each metric's behavior by obtaining their mean scores for 1000 samples. Figures 5.2.1, 5.2.2, and 5.2.3 show the mean metric results for *transformer_summ*, *t5_summ*, and *bart_summ*, respectively.

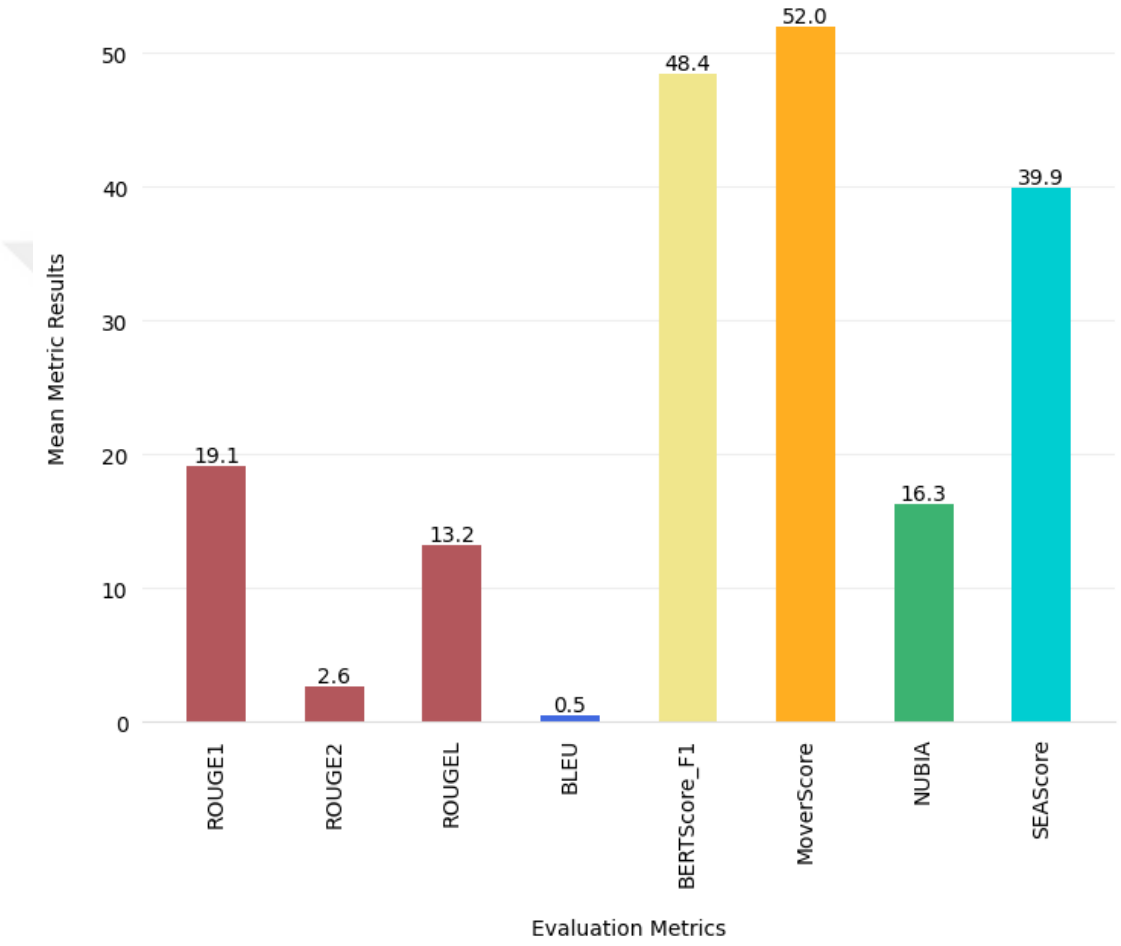

Figure 5.2.1 Mean metric results for *transformer_summ*

Although *transformer_summ* showed good training results after training for 50 epochs, the model may generate summaries with some repetitions and incoherent sentences, as will be evident in some individual output examples provided in Tables 5.5.3, 5.6.5, and 5.6.6. Moreover, word embedding metrics such as BERTScore [18] and MoverScore [29] show higher scores for *transformer_summ* samples compared to n-gram metrics, ROUGE [15] and BLEU [16], and model-based metrics like NUBIA

[31] and our SEAScore metric. The overall mean metric scores for *transformer_summ* vary for each metric.

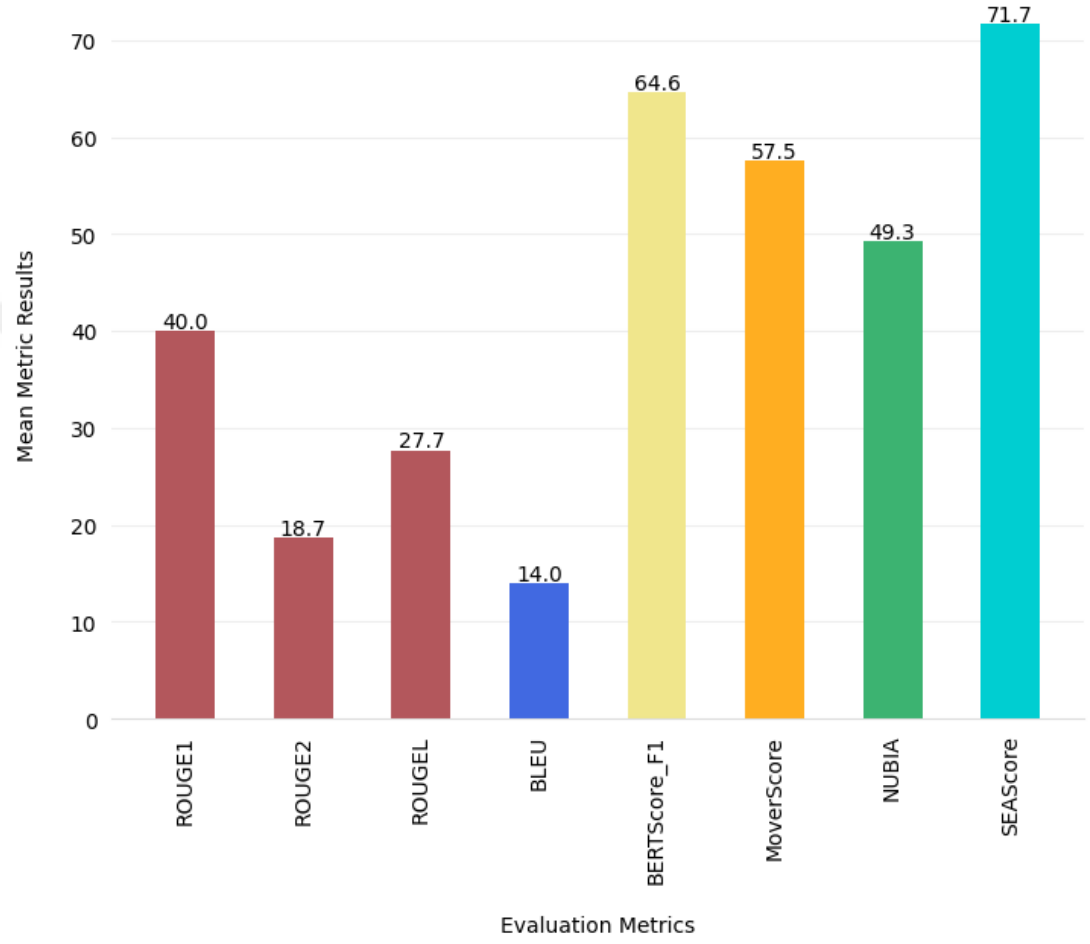

Figure 5.2.2 Mean metric results for *t5_summ*

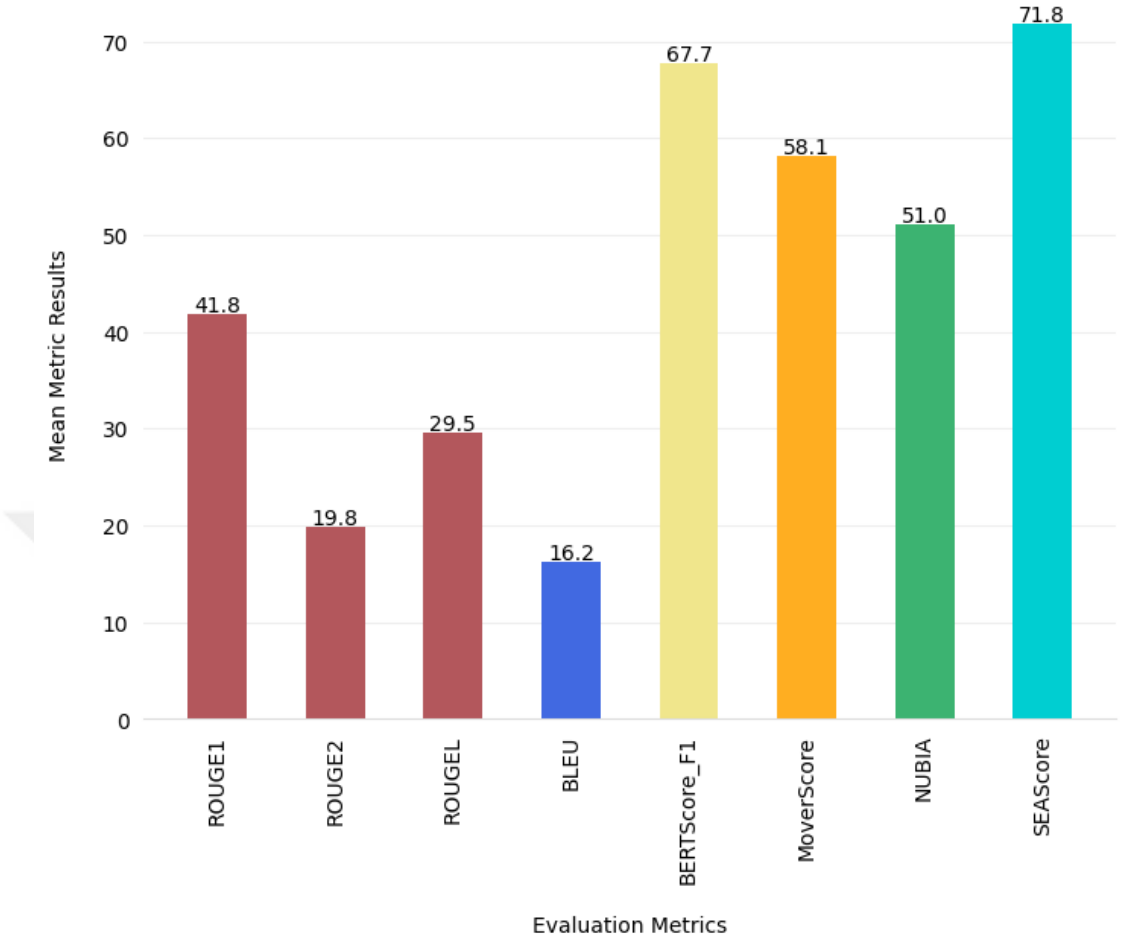

Figure 5.2.3 Mean metric results for *bart_summ*

Metric results for *t5_summ* and *bart_summ* show close results across all metrics, with mean metric scores for *bart_summ* having a slight edge over *t5_summ*. Both models can generate well-made summaries since they are finetuned on pre-trained models that can use the knowledge gained from being trained on large textual corpora, giving *t5_summ* and *bart_summ* an advantage over *transformer_summ*, which was trained from scratch. SEAScore gives higher mean results compared to other metrics. Table 5.2 summarizes the mean metric results for each model.

Table 5.2 Metric results for summarization models

Metrics	**t5_summ**	**bart_summ**	**Transformer_summ**	**Dataset**
ROUGE-1	40.0	41.8	19.1	CNN-DM
ROUGE-2	18.7	19.8	2.6	CNN-DM
ROUGE-L	27.7	29.5	13.2	CNN-DM
BLEU	14.0	16.2	0.5	CNN-DM
BERTScore- F1	64.6	67.7	48.4	CNN-DM
MoverScore	57.5	58.1	52.0	CNN-DM
NUBIA	49.3	51.0	16.3	CNN-DM
SEAScore	71.7	72.8	39.9	CNN-DM

One of the aims of this experiment is to have a preliminary result showing that SEAScore can compare different summarization systems and measure their improvements. From Table 5.2, metric scores indicate that *bart_summ* shows higher metric results compared to *t5_summ* and *transformer_summ*. Overall, all metrics scores and SEAScore show that *bart_summ* is the best-performing summarization model among the three models. This experiment shows that SEAScore is accurately consistent with other standard metrics.

As an observation, it is important to consider how each metric computes its respective scores. ROUGE [15] depends on the number of overlapping n-grams between two summaries, which may not be efficient since it ignores the contextual information of the words in the summaries. On the other hand, BERTScore [18] and MoverScore [29] compute similarity and distance between tokens in candidate and reference summaries using contextualized embeddings obtained from a BERT-based model. However, according to Kaster et al. [73], both BERTScore [18] and MoverScore [29] tend to suffer from sensitivity to lexical adversaries. For example, BERTScore [18] may give an inaccurate score for sentences “man bites dog” vs. “dog bites man” [73]. Consequently, only depending on n-gram units and word embeddings to evaluate summaries may not be enough to cover all aspects of linguistic complexities of

language. The idea behind SEAScore, using the method also used by Kane et al. [31], is to harness the power of pre-trained language models to extract linguistic features. These features offer a broader look at both candidate and reference summaries by examining their semantic relations, level of entailment, and grammatical acceptability. Thus, producing a score that is a cumulation of these linguistic features. We hope to develop an evaluation metric that evaluates summaries similar to how humans approach evaluations by looking into multiple linguistic aspects of a text and deciding on its quality based on linguistic factors.

5.3 SEAScore Metric Results for Summarization Models

In this section, we will use the same 1000 samples from section 5.2 generated from each summarization model, *transformer_summ*, *t5_summ*, and *bart_summ*, to assess how SEAScore evaluates model-generated summaries against their counterpart reference summaries. Figures 5.3.1, 5.3.2, and 5.3.3 show the distribution of SEAScore-generated scores for 1000 test samples for each of the three models.

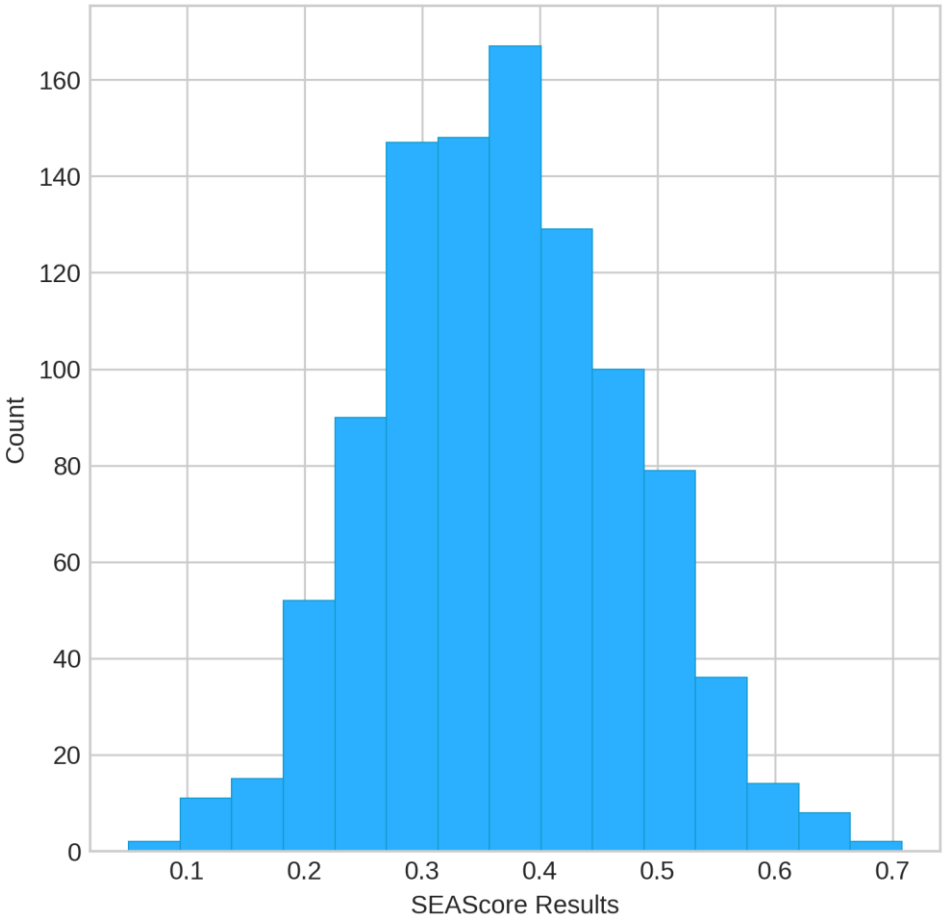

Figure 5.3.1 SEAScore results distribution for *transformer_summ*

Figure 5.3.1 shows that samples evaluated by SEAScore for *transformer_summ* fall between 0.2 and 0.6. The maximum SEAScore score obtained from this sample was 70.77, and the minimum score was 5.06, with a mean SEAScore of 36.92.

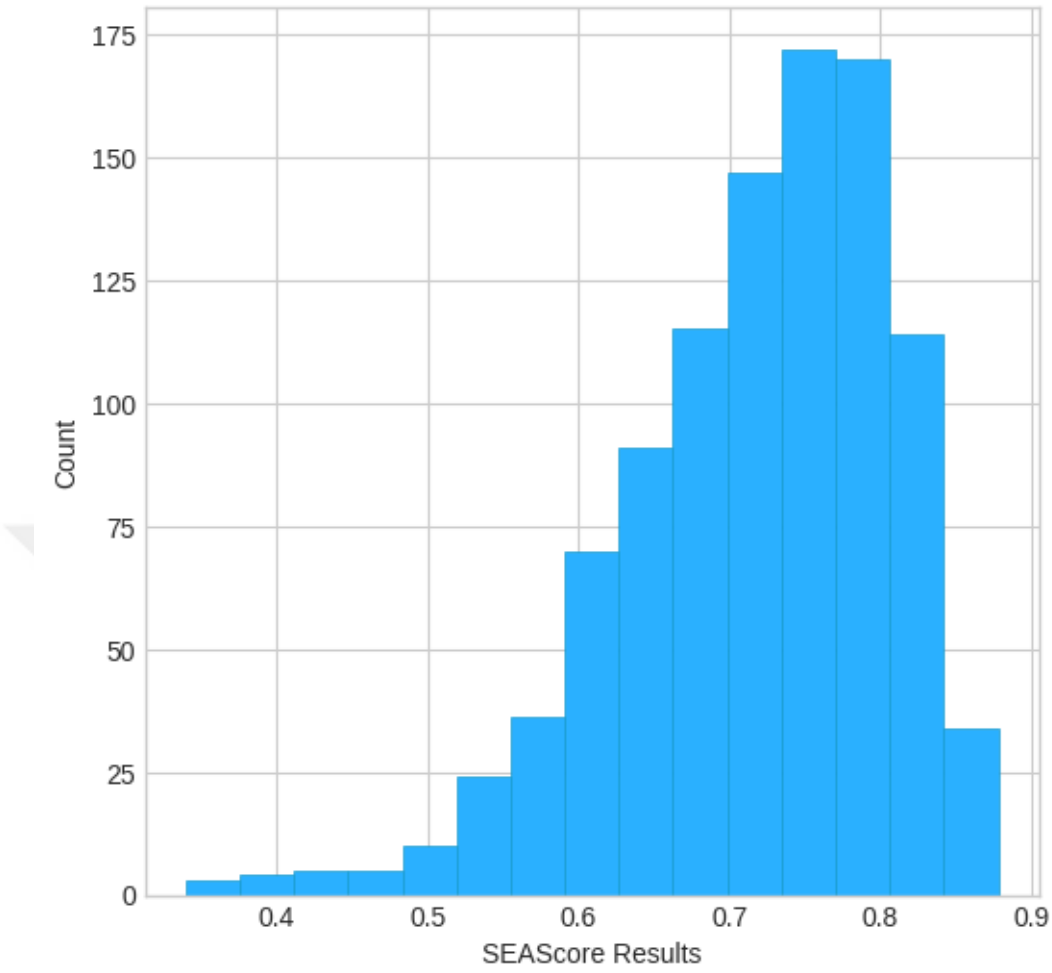

Figure 5.3.2 SEAScore results distribution for *t5_summ*

The distribution for SEAScore results of *t5_summ* is skewed to the right, with most scores being in the range of 0.6 and approximately 0.85. The maximum SEAScore result was 87.82, and the minimum was 33.92, with the mean SEAScore being 71.7.

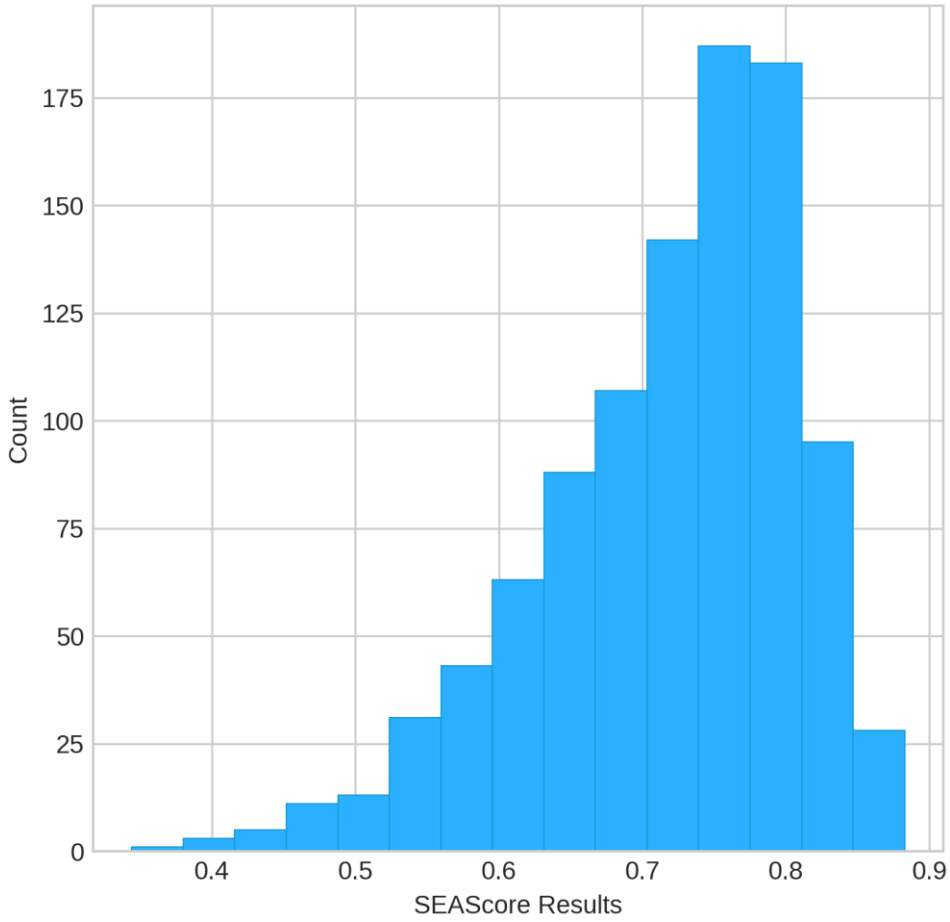

Figure 5.3.3 SEAScore results distribution for *bart_summ*

The distribution for SEAScore results of *bart_summ* is also skewed to the right, with most scores being between 0.6 and approximately 0.85. The maximum SEAScore result was 88.29, and the minimum was 34.43, with the mean SEAScore being 71.8.

Table 5.3 A summary of the mean SEAScore results for each model

Model	**Mean SEAScore Value**	**Sample**	**Dataset**
transformer_summ	39.9	1000	CNN/DailyMail
t5_summ	71.7	1000	CNN/DailyMail
bart_summ	71.8	1000	CNN/DailyMail

Compared to *t5_summ* and *bart_summ*, *transformer_summ* performs worse in model-generated summaries, as can be observed from the metric scores from Table 5.2 and further from the example-generated summaries from sections 5.5 and 5.6. If we look closer at the distribution from Figure 5.3.1, we can see that most SEAScore results are located in the range of 0.3 and 0.5. On the other hand, we can observe from Figure 5.3.2 and Figure 5.3.3 that the SEAScore distributions for *t5_summ* and *bart_summ* are skewed to the right, which indicates an improvement in model-generated summaries obtained from *t5_summ* and *bart_summ* with most SEAScore results for both models being in the range of 0.6 and 0.8.

5.4 Correlation with Human Judgment

This section will experiment with how well our SEAScore metric correlates with human judgments and other automatic evaluation metrics. We will use the REALSumm [56] meta-evaluation tool containing 100 human-annotated examples from the CNN/DailyMail dataset [21]. A Detailed explanation of the REALSumm tool by Bhandari et al. [56] is given in section 3.6.6 of Chapter 3, and the models and metrics we selected to conduct this experiment. The only exception of metric selection is the absence of NUBIA [31] and BLEU [16] since they are not provided in the REALSumm tool. This experiment will be conducted on two levels, system-level evaluation, and summary-level evaluation.

5.4.1 Correlations for top-k Models

This analysis explores how well metrics quantify improvements among summarization models [56]. This system-level analysis measures how well SEAScore correlates with human judgments against other metrics on the outputs of *top-k* abstractive summarization models. The ranking of *top-k* models is done based on the mean human judgment scores for each system; this is measured by equation (3.6.6.3). Moreover, both Pearson and Kendall correlation is computed using equation (3.6.6.2). The correlations are measured between the pairs of human judgment scores, denoted by *litepyramid_recall*, and each automatic evaluation metric. Figures 5.4.1.1 and 5.4.1.2 demonstrate the correlation results for both Kendall and Pearson correlations for systems *k*.

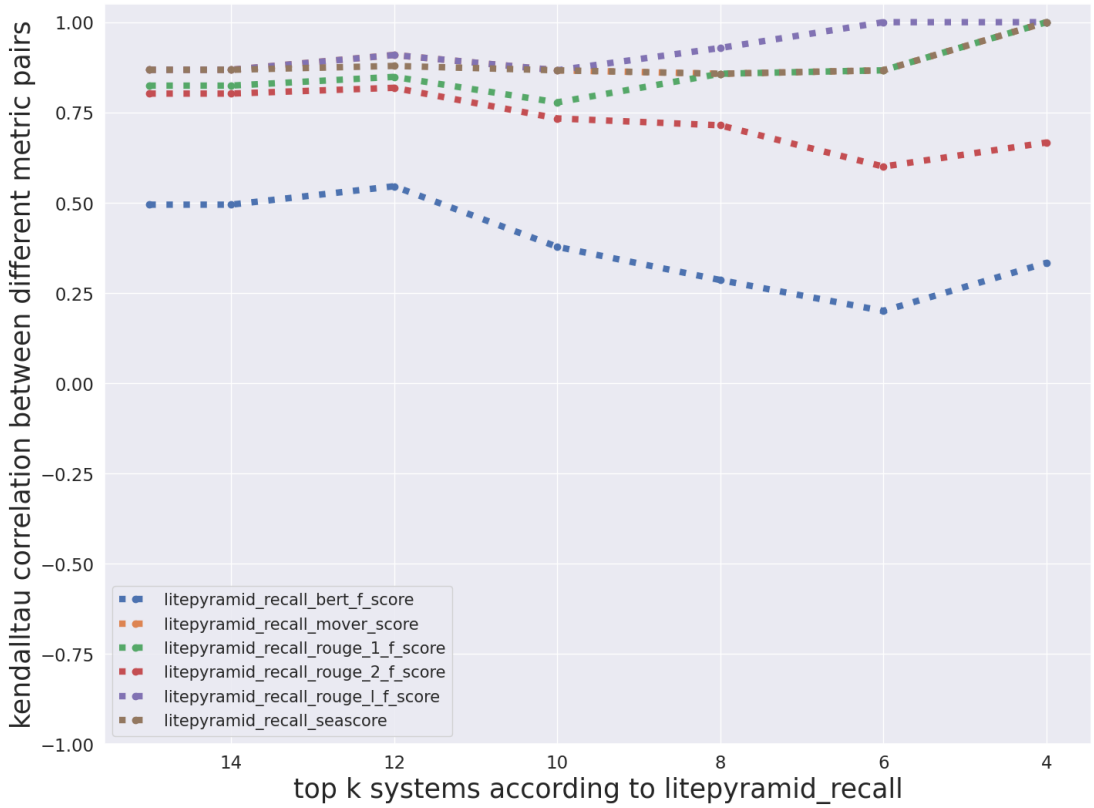

Figure 5.4.1.1 Kendall Correlation between metrics and human judgment scores

From Figure 5.4.1.1, SEAScore shows a higher Kendall correlation with human judgment than BERTScore and ROUGE-2. From $k = 14$ to $k = 8$, SEAScore has a higher Kendall correlation than ROUGE-1 and from $k = 8$ and beyond, SEAScore, ROUGE-1, and MoverScore, show a similar Kendall correlation.

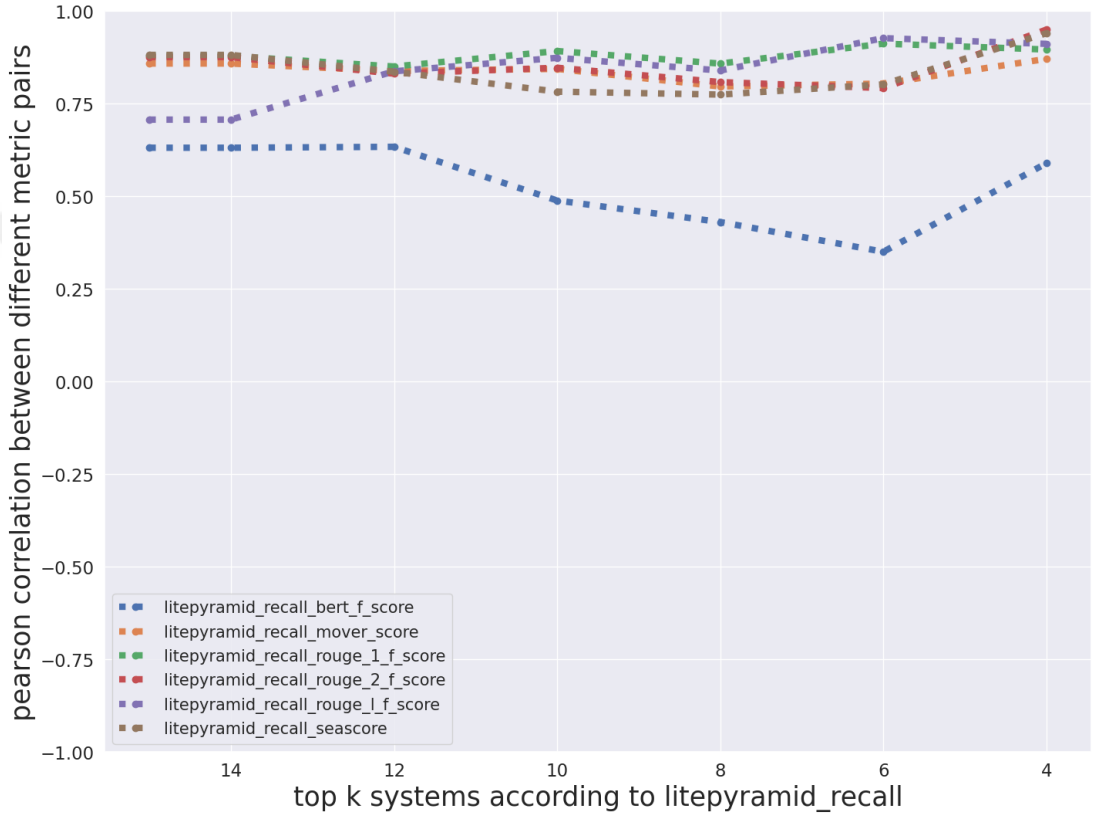

Figure 5.4.1.2 Pearson Correlation between metrics and human judgment scores

From Figure 5.4.1.2, SEAScore shows a higher Pearson correlation than BERTScore, at $k = 14$, SEAScore shows a higher Pearson correlation than ROUGE-L, and at $k = 4$, SEAScore shows higher Pearson correlations than ROUGE-L, ROUGE-1, BERTScore, and MoverScore.

5.4.2 Correlations for a Summary Level

A summary-level analysis measures how well SEAScore correlates with human judgment on individual summaries; this is achieved by computing Kendall and Pearson correlations between different metrics and between metrics and human judgment scores. Similar to the previous section, the human judgment measure is denoted by *litepyramid_recall*, and the correlation computations are carried out by equation (3.6.6.1) in section 3.6.6 of Chapter 3. The following correlation matrices show correlation results obtained for both Kendall and Pearson correlations. The correlation between SEAScore and other metrics will be highlighted.

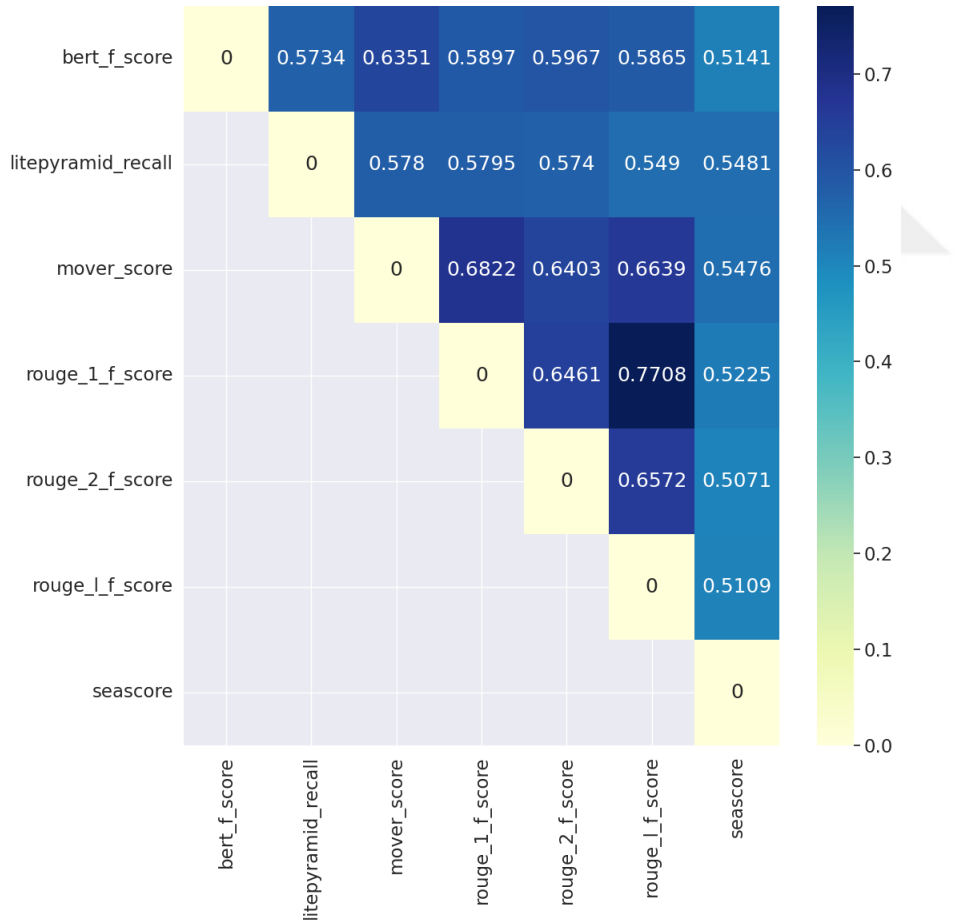

Figure 5.4.2.1 Kendall Correlation for summary level evaluation

From Figure 5.4.2.1, the Kendall correlation indicates that SEAScore closely correlates with human scores on the individual summary level with ROUGE-L.

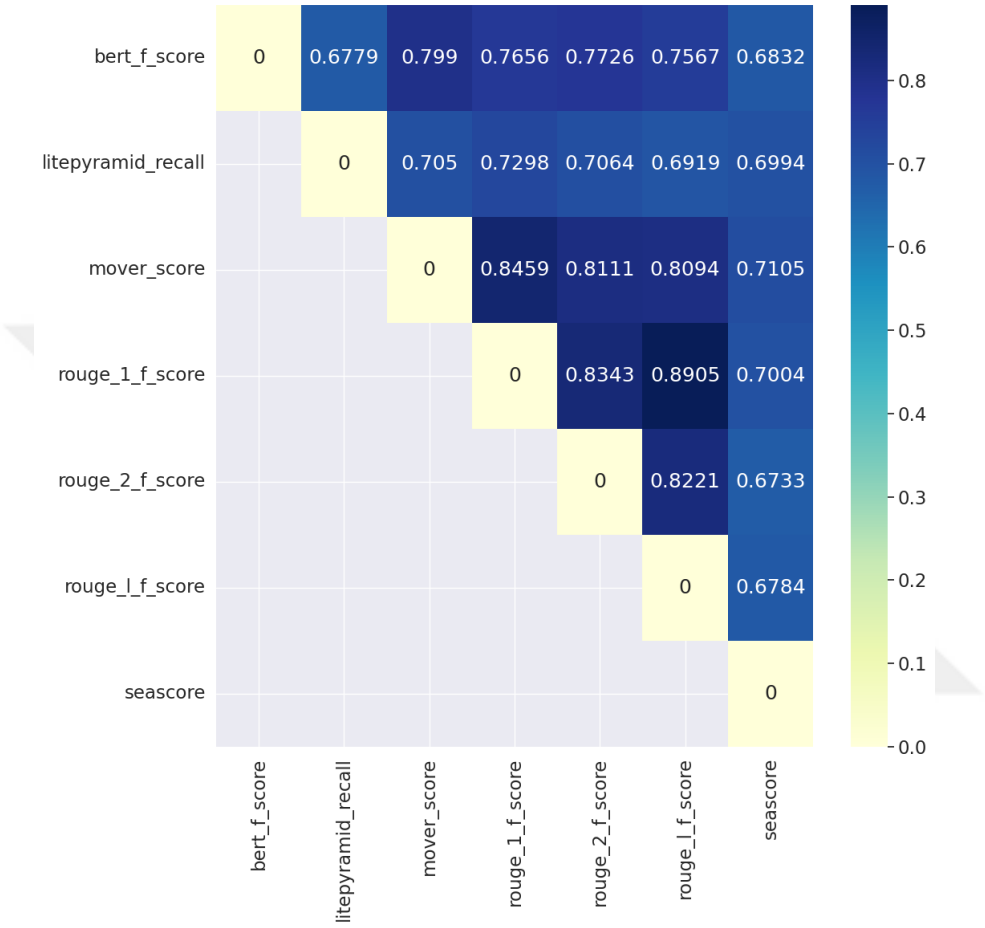

Figure 5.4.2.2 Pearson Correlation for summary level evaluation

From Figure 5.4.2.2, Pearson correlation indicates that SEAScore has a higher summary-level correlation with human judgment compared to BERTScore and ROUGE-L.

5.5 Model Outputs with SEAScore Features

Tables 5.5.1, 5.5.2, and 5.5.3 will fully assess candidate summaries and reference summaries using eight features obtained from our SEAScore metric and its evaluation score. The aim is to take a closer look into how SEAScore conducts the evaluation process. The eight features that SEAScore incorporates are the following: a semantic similarity score extracted by *all-mpnet-base-v2* [62], contradiction, neutral, and entailment scores extracted by *deberta-xlarge-mnli* [63], acceptability or unacceptability scores extracted by *roberta-base-CoLA* [64], and a score produced by a linear regression aggregator using all the features mentioned earlier. It is important to note that the semantic similarity score is the dominant feature, so it heavily contributes to the final SEAScore result. Moreover, the candidate summaries are extracted using our three summarization models with examples from the CNN/DailyMail test set. All features and final scores are within the range of 1-100.

Table 5.5.1 *t5_summ* example

Source document	Ulster have announced the signing of New Zealand international Charles Piutau. Piutau, who can play full back, wing or centre, has agreed a two-year deal from July 2016. He has won 14 caps for the All Blacks, making his Test debut against France two years ago. 'To secure someone of Charles' ability is hugely exciting for us,' Ulster team manager Bryn Cunningham said. New Zealand international Charles Piutau will join Ulster on a two-year deal from 2016 . The 23-year-old has played for the Blues in New Zealand since 2012 and has won 14 caps for the national side . 'Our aim is to produce local players with the ability to play for Ulster and Ireland, and then supplement them with top-quality internationals. 'As he has shown for both the (Auckland) Blues and New Zealand, he has searing pace and great feet. He has an exceptional off-loading game and he is rock-solid in defence. 'He also possesses leadership qualities and maturity beyond his years, and that came across strongly in my conversations with him.
Reference summary	Charles Piutau has played 14 internationals for New Zealand. Piutau can play as full back, wing or centre and will join Ulster in 2016. Ulster team manager Bryn Cunningham is excited to secure Piutau
Model-generated summary	Charles Piutau will join Ulster on a two-year deal from 2016. The 23-year-old has played for the Blues in New Zealand since 2012. Piutau has won 14 caps for the All Blacks, making Test debut against France two years ago.
Semantic similarity	92.05
Contradiction	0.06
Neutral	99.65
Entailment	0.29
Reference unacceptability	2.85
Reference acceptability	97.15
Candidate unacceptability	4.24
Candidate acceptability	95.75
SEAScore result	83.47

In Table 5.5.1, it can be observed that both reference and candidate summaries share the same core information but with different wordings. Thus, there is a 92.05% similarity between them. Regarding other features, the two summaries are neutral with 99.65%, meaning it is undetermined if they entail each other. Both reference and candidate summaries are linguistically acceptable, with 97.15% and 95.75%, respectively. Putting all these features together, the aggregator produces a SEAScore result of 83.47%.

Table 5.5.2 *bart_summ* example

Source document	Relegation-threatened Romanian club Ceahlaul Piatra Neamt have sacked Brazilian coach Ze Maria for the second time in a week. Former Brazil defender Ze Maria was fired on Wednesday after a poor run, only to be reinstated the next day after flamboyant owner Angelo Massone decided to 'give the coaching staff another chance.' But the 41-year-old former Inter Milan and Parma right back, capped 25 times by Brazil, angered Massone again after Ceahlaul were beaten 2-0 by mid-table FC Botosani on Saturday. Ze Maria represented Brazil on 25 occasions during an international career spanning five years . The result left Ceahlaul 16th in the standings, six points adrift of safety. Ze Maria replaced Florin Marin in January to become Ceahlaul's third coach this season. He will be replaced by Serbian Vanya Radinovic.
Reference summary	Romanian club Ceahlaul Piatra Neamt sacked Ze Maria on Wednesday. But the former Brazil international was reinstated the next day. He was then sacked again following a defeat on Saturday.
Model-generated summary	Relegation-threatened Romanian club Ceahlaul Piatra Neamt have sacked Brazilian coach Ze Maria for the second time in a week. The 41-year-old former Inter Milan and Parma right back was sacked on Wednesday.
Semantic similarity	90.99
Contradiction	0.08
Neutral	98.32
Entailment	1.59
Reference unacceptability	7.75
Reference acceptability	92.24
Candidate unacceptability	18.13
Candidate acceptability	81.86
SEAScore result	81.15

In Table 5.5.2, upon inspecting both reference and candidate summaries, it can be observed that both summaries convey similar information, though the word order may differ. There is a 90.99% similarity between them, and the summary pairs are highly neutral with 98.32% with slight entailment of 1.59% entailment. Regarding linguistic acceptability, the reference summary is more acceptable, with 92.24%, compared to the acceptability of the candidate summary, which is 81.86%. So, the final SEAScore result is 81.15%.

Table 5.5.3 *transformer_summ* example

Source document	Firefighters responded to cries for help - from two parrots. The crew scoured a burning home in Boise, Idaho, searching for people shouting 'Help!' and 'Fire!' Eventually, to their surprise, they found a pair of squawking birds. SCROLL DOWN FOR VIDEO . Cry for help! This is one of the two parrots who were found in a burning home after calling for help . The tropical creatures appeared to have been alone when flames began to sweep the property. But they seemed to know what to do. Both were pulled from the home and given oxygen. They are expected to survive. The fire crew in Boise, Idaho, thought they were chasing human voices when the found the birds . Treatment: The officials treated the birds with oxygen masks and both are expected to survive . According to KBOI, the cause of the officers managed to contain the fire to just one room. It is being investigated and no people were found inside. Officials have yet to track down the birds' owners.
Reference summary	Two parrots were home alone when a fire erupted in Boise, Idaho. Started calling 'Help!' and 'Fire!', crew thought they were human voices. Both were pulled from the wreckage and treated with oxygen masks.
Model-generated summary	a woman and the woman were involved in a search of a nearby nearby nearby residents and have been at a nearby mall police say it is not known
Semantic similarity	18.50
Contradiction	25.72
Neutral	73.93
Entailment	0.34
Reference unacceptability	91.22
Reference acceptability	8.78
Candidate unacceptability	61.38
Candidate acceptability	38.61
SEAScore result	21.64

In Table 5.5.3, by examining both reference and candidate summaries, it is clear that the candidate summary suffers from linguistic problems, and the two summaries have no information in common. The metric extracted a similarity score of 18.50%, with contradiction and neutral scores of 25.75% and 73.93%, respectively. Moreover, reference and candidate summaries are more linguistically unacceptable, with 91.22% and 61.38% unacceptability scores, respectively. Using the information provided by the features, the aggregator gives a SEAScore of 21.64%, indicating that the quality of the overall candidate summary is insufficient.

5.6 Model Outputs with Metric Results

Tables 5.6.1, 5.6.2, 5.6.3, 5.6.4, 5.6.5, and 5.6.6 demonstrate random examples from the CNN/DailyMail [21] dataset on the following few pages. These examples include the source article, reference summary, model-generated summary, and metric scores for each sample. The phrases model-generated summary and candidate summary will be used interchangeably. We obtained the candidate summaries using *t5_summ*, *bart_summ*, and *transformer_summ* models. The aim is to take a closer look at metric scores for individual summary examples. We will share our observations for each example.

Table 5.6.1 *t5_summ* example 1

Source document	Chelsea are looking to beat Manchester City to sign Brazilian prospect Nathan. The attacking midfielder turned 19 last month but has been in contract dispute with his club Atletico Paranaense. He is due to speak to Chelsea next week ahead of a proposed move to Stamford Bridge which would likely see him loaned out. Atletico Paranaense attacking midfielder Nathan is attracting interest from Chelsea and Manchester City . Jose Mourinho and Manuel Pellegrini will go head-to-head to sign Brazilian starlet Nathan . Manchester City have been keen on Nathan following his performances in the World U17 Championships two years ago and he has since broken into Paranaense's first team. Chelsea have a strong Brazilian contingency with the likes of Willian, Ramires, Filipe Luis and Oscar while 21-year-old Lucas Piazon is on loan at Eintracht Frankfurt. Nathan would join fellow Brazilians Willian, Ramires, Filipe Luis and Oscar (pictured) at Stamford Bridge.
Reference summary	Both Chelsea and Manchester City are keen on signing Nathan. The attacking midfielder has been in contract dispute with his current club. Nathan is due to speak to Chelsea next week ahead of proposed move.
Model-generated summary	Atletico Paranaense attacking midfielder Nathan is attracting interest from Chelsea and Manchester City. The 19-year-old turned 19 last month but has been in contract dispute with Atletico Paranaense. He is due to speak to Chelsea next week ahead of a proposed move to Stamford Bridge.
ROUGE-1	63.41
ROUGE-2	50
ROUGE-L	56.10
BLEU	36.09
BERTScore-F1	80.37
MoverScore	63.48
NUBIA	69.68
SEAScore	76.60

In Table 5.6.1, it can be observed that both the reference summary and the candidate model-generated summary give the same core information. However, the model-generated summary has repetitive information, such as “the 19-year-old turned 19 last month”. So BERTScore -F1 over-scores compared to SEAScore.

Table 5.6.2 *t5_summ* example 2

Source document	Cristiano Ronaldo returned to top form last weekend with five goals against Granada, and he is now urging his supporters to get themselves into shape. The Real Madrid forward has not been at his best this year, but turned things around with a stunning display on Sunday, during Real Madrid's 9-1 win. And the Portuguese star, and World Player of the Year, took to Twitter to share his celebrations with his many followers, posing with a bike and encouraging them to start riding. Cristiano Ronaldo scored five goals as Real Madrid thrashed Granada 9-1 on Sunday afternoon . Ronaldo is one of the fittest athletes in the world, and tweeted for his many fans to join him in using excercise as a way of feeling better. 'Exercise all you can!' he wrote on Twitter. 'It's good for your body and your mind!' The Portuguese superstar was simply irresistible as Real ran riot at the Bernabeu .
Reference summary	Cristiano Ronaldo scored five times in Real Madrid's 9-1 win over Granada. Ronaldo posts picture with a bike on Twitter. Real Madrid star tells followers 'Exercise all you can!'READ: Which clubs have suffered most at the hands of the Ronaldo?CLICK HERE for all the latest Real Madrid news .
Model-generated summary	Cristiano Ronaldo scored five goals as Real Madrid thrashed Granada 9-1 on Sunday .The Portuguese forward has not been at his best this year, but turned things around with a stunning display .Ronaldo tweeted for his fans to join him in using excercise as way of feeling better .'Exercise all you can!' he wrote on Twitter .'It's good for your body and your mind!' he wrote on Twitter .'It's good for your body and your mind!'
ROUGE-1	36.64
ROUGE-2	15.50
ROUGE-L	22.90
BLEU	10.67
BERTScore-F1	62.31
MoverScore	56.42
NUBIA	43.51
SEAScore	75.22

In Table 5.6.2, the model-generated summary gives more accurate information compared to the reference summary, but it repeats the phrase "It's good for your body and your mind" twice. Overall, the model-generated summary gives a better context of the topic than the reference summary, so the SEAScore result better represents the quality of the candidate summary.

Table 5.6.3 *bart_summ* example 1

Source document	A pit crew member was hit by a car on Sunday during the inaugural IndyCar Grand Prix of Louisiana. Todd Phillips, a front-outside tire changer for Dayle Coyne Racing, was injuried when he was struck by the car of Francesco Dracone, who had come in on Lap 25 for tires and fuel. Dracone spun while exiting his put box, clipping Phillips' leg. Tire changer Todd Phillips, a front-outside tire changer for Dayle Coyne Racing, was injuried when he was struck by the car of Francesco Dracone, who had come in on Lap 25 for tires and fuel . Phillips was taken to the infield care center for treatment where he has received stitches for a cut on his leg and has been released. Many cars struggled with traction during the race after rain had fallen for hours leading up to the green flag. Dracone did not finish the race and wound up 23rd. Francesco Dracone (19), of Italy, spins out on turn one during the IndyCar Grand Prix of Louisiana auto race in Avondale on Sunday .
Reference summary	Pit crew member Todd Phillips was hit by a car on Sunday during the inaugural IndyCar Grand Prix of Louisiana. He was injuried when he was struck by the car of Francesco Dracone, who had come in on Lap 25 for tires and fuel. Phillips received stitches for a cut on his leg and has been released. Dracone did not finish the race and wound up 23rd.
Model-generated summary	Todd Phillips, a front-outside tire changer for Dayle Coyne Racing, was injuried when he was struck by the car of Francesco Dracone. Dracone spun while exiting his put box, clipping Phillips' leg. Phillips was taken to the infield care center for treatment where he has received stitches for a cut on his leg and has been released.
ROUGE-1	52.80
ROUGE-2	39.02
ROUGE-L	44.80
BLEU	35.18
BERTScore-F1	72.00
MoverScore	58.87
NUBIA	56.25
SEAScore	77.06

In Table 5.6.3, both reference and candidate summaries give similar core information. So both BERTScore-F1 and SEAScore give accurate assessments of the summaries.

Table 5.6.4 *bart_summ* example 2

Source document	London Welsh have announced a contract extension for their former England international back Olly Barkley. The 33-year-old joined the Exiles last year, having previously played for Bath, Gloucester, Racing Metro, Grenoble and the Scarlets. He won 23 Test caps, the last of which was against New Zealand in 2008. Olly Barkley has extended his stay at London Welsh after signing a new contract at the relegated-club . Welsh are set to return to the Championship next season after finishing bottom of the Aviva Premiership. 'It has not been an easy year, but I am committed to seeing my journey with London Welsh through,' Barkley said. 'I am really looking forward to working with our new team and the reinvigorated coaching regime, which is being put together, to put this season behind us and lay a strong foundation for promotion back to the Premiership for the 2016-17 season.' The fly half featured 23 times for England and previously played for Bath, Gloucester and Racing Metro .
Reference summary	Olly Barkley has signed a contract extension at London Welsh. The Exiles were relegated from the Aviva Premiership this season. Fly half Barkley is keen to fire the Welsh back up to the topflight. CLICK HERE for all the latest rugby unions news.
Model-generated summary	London Welsh have announced a contract extension for Olly Barkley. The 33-year-old has previously played for Bath, Gloucester, Racing Metro, Grenoble and the Scarlets. Barkley won 23 Test caps, the last of which was against New Zealand in 2008.
ROUGE-1	30.59
ROUGE-2	9.64
ROUGE-L	16.47
BLEU	0.0
BERTScore-F1	58.73
MoverScore	54.66
NUBIA	42.61
SEAScore	79.01

In Table 5.6.4, the reference and candidate summaries focus on different information from the source documents while retaining the same main topic. The candidate summary is easier to read in terms of phrasing.

Table 5.6.5 *transformer_summ* example 1

Source document	Stoke City are closing on Standard Liege starlet Thibaut Verlinden. The 15-year-old midfielder is a highly regarded member of Belgium's U16 squad and has had trials with Liverpool and Everton. Club Brugge and Anderlecht have also expressed an interest but Verlinden, whose father Dany used to be a goalkeeper, is poised to sign on July 9 when he turns 16. Thibaut Verlinden (right), in action for Belgium against England U16s, looks to be joining Stoke City . Stoke manager Mark Hughes has also persuaded Barcelona's 19-year-old winger Mohamed El Ouriachi to join this summer. Hughes said: 'We are trying to improve the quality of the under-21s to get more young players knocking on the door of the first-team squad. That's our intention.' Stoke retain an interest in Sunderland's Lee Cattermole whose contract talks have been put on hold until their Premier League fate is resolved. Mohamed El Ouriachi competes for the ball for Barcelona, and is set to join Stoke this summer .
Reference summary	Belgian 15-year-old star has had trials at Liverpool and Everton. Thibaut Verlinden will join Stoke when he turns 16 in July .Mark Hughes will also sign Moha El Ouriachi from Barcelona this summer.
Model-generated summary	manuel pellegrini said he could be keen on a new deal the pair to make a summer move for sunderland and is set to sign for a new deal the 24yearold and have made more than their premier league debut last season on sunday.
ROUGE-1	9.88
ROUGE-2	0.0
ROUGE-L	4.34
BLEU	0.0
BERTScore-F1	53.26
MoverScore	52.89
NUBIA	18.04
SEAScore	44.61

Table 5.6.5 shows an apparent contradiction between the reference and candidate summaries regarding core information. Moreover, the candidate summary has linguistic and grammatical issues. So BERTScore-F1 and MoverScore overscore compared to SEAScore.

Table 5.6.6 *transformer_summ* example 2

Source document	St Etienne want to sign Cardiff full-back Kevin Theophile-Catherine on a permanent deal. The 25-year-old signed for Cardiff from Stade Rennais for £2.1million but has been on loan with St Etienne this season. They have an option to make the deal permanent for £1.5million but Theophile-Catherine wants to see if there are other options before committing. He has two years left on contract at Cardiff. St Etienne defender Kevin Theophile-Catherine shields the ball from PSG forward Zlatan Ibrahimovic . St Etienne, meanwhile, will not take up an option to sign Norwich striker Ricky Van Wolfswinkel on a permanent deal. The Dutchman scored only one goal following his £8.5million move to Norwich from Sporting Lisbon and has scored six times this season for St Etienne in 31 appearances. He will return to Carrow Road at the end of the season. Theophile-Catherine (right) competes for a header with Montpellier forward Kevin Berigaud in December .
Reference summary	The 25-year-old signed for Cardiff from Stade Rennais for £2.1million. But the defender has been on loan with St Etienne this season. The club have an option to make the deal permanent for £1.5million. St Etienne will not take up an option to sign Norwich striker Ricky Van Wolfswinkel on a permanent deal.
Model-generated summary	the german side are keen on the clubs target for the club the french champions have suggested the midfielder talks with the new manager of the club the former fulham and west brom have not been in the past
ROUGE-1	22.45
ROUGE-2	2.08
ROUGE-L	16.33
BLEU	0.0
BERTScore-F1	44.83
MoverScore	52.04
NUBIA	6.66
SEAScore	33.92

In Table 5.6.6, the candidate summary is ambiguous in terms of information and grammatical accuracy, so the SEAScore and NUBIA results are more acceptable than the BERTScore-F1 and MoverScore results.

5.7 Limitations and Discussions

The limitations of SEAScore are mainly related to the LLMs that it uses. Although SEAScore incorporates LLMs that achieve state-of-the-art in their respective tasks, better-performing LLMs will boost SEAScore's evaluation performance. Another notable limitation of SEAScore is the linear regression aggregator component. We created a custom dataset containing manually obtained evaluation scores, but the number of samples was limited, with only 151. This limited sample size was due to limited resources for this study. Thus, a larger sample size for the custom dataset would enhance the performance of the aggregator component in SEAScore.

Regarding scalability, a significant advantage of SEAScore is its flexibility, where we add or further train a particular component. This flexible nature can enable us to incorporate more LLMs that add different perspectives to the evaluation process. However, adding more LLMs would require an improved or entirely different aggregation process to accommodate the new LLMs.

Lastly, the summarization field needs better datasets and benchmarks encompassing various human judgments and evaluations. Human judgment is an essential aspect in evaluating summarization models, so having datasets that contain high-quality human judgment scores would help in computing more accurate correlation analysis studies that would give us a more comprehensive view of the performance of automatic evaluation metrics.

CHAPTER 6

CONCLUSION

In this study, we proposed SEAScore, a new automatic evaluation metric for evaluating abstractive text summarization models. It is a model-based metric that utilizes the power of LLMs to extract meaningful linguistic features. SEAScore uses these features to determine the quality of a model-generated candidate summary against its counterpart reference summary. SEAScore measures the quality of a candidate summary using three NLP tasks: semantic similarity, MNLI, and CoLA. Finally, all features are fed into a linear regression aggregator trained on a custom dataset to produce an evaluation score. Moreover, we trained three summarization models. We trained a Transformer model from scratch and finetuned two LLMs, T5 and BART, on the summarization task. All models were trained on the CNN/DailyMail dataset.

We generated a total sample of 3000 test summaries from our models and conducted experiments that compared the performance of SEAScore against other metrics. Furthermore, we conducted a meta-evaluation experiment to measure how well SEAScore correlates with human judgments compared to other metrics. Experimental results show that SEAScore gives a broader evaluation of summaries compared to n-gram and word-embedding metrics since it uses multiple linguistic features and applies a model-based approach to evaluation. Moreover, SEAScore matched some metrics on human judgment correlation and, in some cases, showed better correlation with human judgment than metrics such as BERTScore and ROUGE on evaluating different summarization systems and individual summaries.

As a future direction, metrics should also focus on enhancing the detection of hallucinations and false information generated by abstractive summarization models.

This study primarily focused on evaluating similarity and linguistic acceptability, but future work can focus on adding additional features to check the hallucination and factual consistency of model-generated summaries.

REFERENCES

[1] M. Allahyari *et al.*, “Text Summarization Techniques: A Brief Survey,” *International Journal of Advanced Computer Science and Applications*, vol. 8, pp. 397–405, Jul. 2017.

[2] R. Nallapati, B. Zhou, C. dos Santos, C. Gulcehre, and B. Xiang, "Abstractive text summarization using sequence-to-sequence RNNs and beyond," In *Proc. 20th SIGNLL Conf. Comput. Natural Lang. Learn*, 2016, pp. 280–290.

[3] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *arXiv preprint arXiv:1409.0473*, 2015.

[4] A. M. Rush, S. Chopra, and J. Weston, "A Neural Attention Model for Sentence Summarization," In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 379–389.

[5] A. See, P. J. Liu, and C. D. Manning, "Get To The Point: Summarization with Pointer-Generator Networks," In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1073–1083.

[6] A. Vaswani *et al.*, "Attention is all you need," In *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[7] S. Merity, N. S. Keskar, and R. Socher, "Regularizing and Optimizing LSTM Language Models," *arXiv preprint arXiv: 1708.02182*, 2018.

[8] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, (2019, February), "Language Models are Unsupervised Multitask Learners," *OpenAI blog*, [Online], 1(8), p.9. Available at: https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf [Feb. 19, 2023].

[9] T. Brown *et al.*, "Language Models are Few-Shot Learners," In *Proceedings of Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, 2020, pp. 1877-1901.

[10] T. L. Scao *et al.*, "BLOOM: A 176B-Parameter Open-Access Multilingual Language Model," *arXiv preprint arXiv:2211.05100*, 2022.

[11] C. Raffel *et al.*, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," *Journal of Machine Learning Research*, vol. 21, pp. 1–67, Jan. 2020.

[12] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7871–7880.

[13] J. Zhang, Y. Zhao, M. Saleh, and P. Liu, "PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization*."* In *Proceedings of the 37 the International Conference on Machine Learning*, 2020, pp. 11328–11339.

[14] Z. Cao, F. Wei, W. Li, and S. Li, "Faithful to the original: fact-aware neural abstractive summarization," In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'18/IAAI'18/EAAI'18)*, 2018, pp. 4784–4791.

[15] C. Yew Lin, "ROUGE: A package for automatic evaluation of summaries," In *Text summarization branches out*, 2004, pp. 74–81.

[16] K. Papineni, S. Roukos, T. Ward, and W. Jing Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation," In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 2002, pp. 311–318.

[17] S. Banerjee and A. Lavie, "METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments," In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, 2005, pp. 65–72.

[18] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating Text Generation with BERT," *arXiv preprint arXiv:1904.09675*, 2020.

[19] J. Devlin, M. Wei Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.

[20] "DUC 2004." Internet: https://duc.nist.gov/duc2004/ [Oct. 28, 2022].

[21] Abisee., "cnn-dailymail," Internet: https://github.com/abisee/cnn-dailymail [Aug. 29, 2022].

[22] Y. C. Chen and M. Bansal, "Fast abstractive summarization with reinforce-selected sentence rewriting," In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 675–686.

[23] S. Gehrmann, Y. Deng, and A. Rush, "Bottom-Up Abstractive Summarization," In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 4098–4109.

[24] W. Kryściński, R. Paulus, C. Xiong, and R. Socher, "Improving Abstraction in Text Summarization," In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 1808–1817.

[25] S. Hochreiter, and J. Schmidhuber," Long Short-Term Memory," *Neural computation*, vol. 9, pp. 1735–1780, Nov. 1997.

[26] S. Narayan, S. B. Cohen, and M. Lapata, "Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization," In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 1797–1807.

[27] Y. Liu and M. Lapata, "Text Summarization with Pretrained Encoders," In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3730–3740.

[28] S. Gabriel, A. Bosselut, J. Da, A. Holtzman, J. Buys, K. Lo, A. Celikyilmaz, and Y. Choi, "Discourse Understanding and Factual Consistency in Abstractive Summarization," In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021, pp. 435-447.

[29] W. Zhao, M. Peyrard, F. Liu, Y. Gao, C. M. Meyer, and S. Eger, "MoverScore: Text Generation Evaluating with Contextualized Embeddings and Earth Mover Distance," In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 563–578.

[30] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From Word Embeddings To Document Distances," In *Proceedings of the 32nd International Conference on Machine Learning*, 2015, pp. 957-966.

[31] H. Kane, M. Y. Kocyigit, A. Abdalla, P. Ajanoh, and M. Coulibali, "NUBIA: NeUral Based Interchangeability Assessor for Text Generation," In *Proceedings of the 1st Workshop on Evaluating NLG Evaluation*, 2020, pp. 28-37.

[32] M. Peyrard, T. Botschen, and I. Gurevych, "Learning to Score System Summaries for Better Content Selection Evaluation.," In *Proceedings of the Workshop on New Frontiers in Summarization*, 2017, pp. 74–84.

[33] E. Clark, A. Celikyilmaz, and N. A. Smith, "Sentence Mover's Similarity: Automatic Evaluation for Multi-Sentence Texts,". In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 2748–2760.

[34] T. Scialom, S. Lamprier, B. Piwowarski, and J. Staiano, "Answers Unite! Unsupervised Metrics for Reinforced Summarization Models," In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3246–3256.

[35] O. Vasilyev, V. Dharnidharka, and J. Bohannon, "Fill in the BLANC: Human-free quality estimation of document summaries," In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, 2020, pp. 11–20.

[36] Y. Gao, W. Zhao, and S. Eger, “SUPERT: Towards New Frontiers in Unsupervised Evaluation Metrics for Multi-Document Summarization,” In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 1347–1354.

[37] W. Yuan, G. Neubig, and P. Liu, “BARTScore: Evaluating Generated Text as Text Generation.” *Advances in Neural Information Processing Systems 34 (NeurIPS 2021)*, vol. 34, pp. 27263–27277, Oct. 2021.

[38] D. Chandrasekaran and V. Mago, “Evolution of Semantic Similarity—A Survey,” *ACM Computing Surveys*, vol. 54, pp. 1–37, Feb. 2021.

[39] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representation in Vector Space," *arXiv preprint arXiv:1301.3781*, 2013.

[40] A. L. Kalouli, A. Buis, L. Real, M. Palmer, and V. de Paiva, "Explaining Simple Natural Language Inference," In *Proceedings of the 13th Linguistic Annotation Workshop*, 2019, pp. 132–143.

[41] F. Chollet, *Deep Learning with Python 2nd edition*, Manning Publication Co, 2021, pp. 347-348.

[42] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," *arXiv preprint arXiv:1908.10084*, 2019.

[43] Reimers, "SentenceTransformers Documentation," Internet: https://github.com/UKPLab/sentence-transformers [Feb. 19, 2023].

[44] Y. Liu *et al.* "RoBERTa: A Robustly Optimized BERT Pretraining Approach," *arXiv preprint arXiv:1907.11692*, 2019.

[45] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "XLNet: generalized autoregressive pretraining for language understanding," In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, pp. 5753–5763.

[46] P. He, X. Liu, J. Gao, and W. Chen, "DeBERTa: Decoding-enhanced BERT with Disentangled attention," *arXiv preprint arXiv:2006.03654*. 2021.

[47] K. Purohit, "Understanding how the XLNet outperforms BERT in Language Modeling." Internet: https://medium.com/saarthi-ai/xlnet-the-permutation-language-model-b30f5b4e3c1e [Oct. 09, 2022].

[48] K. Song, X. Tan, T. Qin, J. Lu, and T. Yan Liu, "MPNet: masked and permuted pre-training for language understanding," In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS'20)*, 2020, pp. 16857–16867.

[49] X. Tan, "MPNet combines strengths of masked and permuted language modeling for language understanding." Internet: https://www.microsoft.com/en-us/research/blog/mpnet-combines-strengths-of-masked-and-permuted-language-modeling-for-language-understanding/ [Oct. 09, 2022].

[50] A. R. Fabbri, W. Kryscinski, B. McCann, C. Xiong, R. Socher, and D. R. Radev, "SummEval: Re-evaluating Summarization Evaluation," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 391–409, Apr. 2021.

[51] Fabbri., "Summarization Repository," Internet: https://github.com/Yale-LILY/SummEval [Oct. 10, 2022].

[52] P. May, "Machine translated multilingual STS benchmark dataset," Internet: https://github.com/PhilipMay/stsb-multi-mt [Oct. 08, 2022].

[53] A. Williams, N. Nangia, and S. Bowman, "The Multi-Genre NLI Corpus," Internet: https://cims.nyu.edu/~sbowman/multinli/ [Oct. 8, 2020].

[54] A. Williams, N. Nangia, and S. Bowman, "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference," In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018, pp. 1112–1122.

[55] O. Bojar, Y. Graham, and A. Kamran, "Results of the WMT17 Metrics Shared Task," In *Proceedings of the Second Conference on Machine Translation*, 2017, pp. 489–513.

[56] M. Bhandari, P. N. Gour, A. Ashfaq, P. Liu, and G. Neubig, "Re-evaluating Evaluation in Text Summarization," In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 9347–9359.

[57] O. Shapira, D. Gabay, Y. Gao, H. Ronen, R. Pasunuru, M. Bansal, Y. Amsterdamer, and I. Dagan, "Crowdsourcing Lightweight Pyramids for Manual Summary Evaluation," In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 682–687.

[58] L. Dong *et al.*, "Unified Language Model Pre-training for Natural Language Understanding and Generation," *arXiv preprint arXiv:1905.03197*, 2019.

[59] H. Zhang, J. Cai, J. Xu, and J. Wang, "Pretraining-Based Natural Language Generation for Text Summarization," In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, 2019, pp. 789–797.

[60] W. Yoon, Y. S. Yeo, M. Jeong, B. Yi, and J. Kang, "Learning by Semantic Similarity Makes Abstractive Summarization Better," *arXiv preprint arXiv:2002.07767*, 2021.

[61] A. Warstadt, A. Singh, and S. R. Bowman, "Neural Network Acceptability Judgments," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 625-641, Sep. 2019.

[62] Reimers, "sentence-transformers/all-mpnet-base-v2," Internet: https://huggingface.co/sentence-transformers/all-mpnet-base-v2 [Nov. 23, 2022].

[63] P. He, X. Liu, J. Gao, and W. Chen, "microsoft/deberta-xlarge-mnli," Internet: https://huggingface.co/microsoft/deberta-xlarge-mnli [Nov. 23, 2022].

[64] Morris, "textattack/roberta-base-CoLA," Internet: https://huggingface.co/textattack/roberta-base-CoLA [Nov. 23, 2022].

[65] T. Wolf, "HuggingFace," Internet: https://github.com/huggingface/transformers [Nov. 23, 2022].

[66] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *JMLR 12*, vol. 12, pp. 2825-2830, Oct. 2011.

[67] Google Brain Team, "TensorFlow,"
Internet: https://github.com/tensorflow/tensorflow [Nov. 26, 2022].

[68] A. Paszke, S. Gross, S. Chintala, and G. Chanan, "PyTorch," Internet: https://github.com/pytorch/pytorch [Nov. 24, 2022].

[69] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv arXiv:1412.6980*, 2017.

[70] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, pp.1929–1958, Jun. 2014.

[71] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," *arXiv arXiv:1711.05101*, 2019.

[72] S. Yang, "Deep learning basics — weight decay,"
Internet: https://medium.com/analytics-vidhya/deep-learning-basics-weight-decay-3c68eb4344e9 [Nov. 25, 2022].

[73] M. Kaster, W. Zhao, and S. Eger "Global Explainability of BERT-Based Evaluation Metrics by Disentangling along Linguistic Factors," In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 8912–8925.