Pontifícia Universidade Católica
do Rio de Janeiro

Lucas Roberto da Silva

Unsupervised multi-review summarization using fine-tuned Transformer language models

Dissertação de Mestrado

Dissertation presented to the Programa de Pós-graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática.

Advisor: Prof. Sérgio Colcher

Rio de Janeiro
May 2021

PUC-Rio - Certificação Digital Nº 1820996/CA

PONTIFÍCIA UNIVERSIDADE CATÓLICA
DO RIO DE JANEIRO

Lucas Roberto da Silva

Unsupervised multi-review summarization using fine-tuned Transformer language models

Dissertation presented to the Programa de Pós-graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática. Approved by the Examination Committee.

PUC-Rio - Certificação Digital Nº 1820996/CA

Prof. Sérgio Colcher
Advisor
Departamento de Informática – PUC-Rio

Prof. Edward Hermann Haeusler
Departamento de Informática – PUC-Rio

Prof. Helio Côrtes Vieira Lopes
Departamento de Informática – PUC-Rio

Rio de Janeiro, May 14th, 2021

All rights reserved.

Lucas Roberto da Silva

Graduated in Mathematics at Fluminense Federal University in 2017.His main research interests are Natural Language Processing and Deep Learning.

PUC-Rio - Certificação Digital Nº 1820996/CA

Bibliographic data

Silva,Lucas Roberto da

Unsupervised multi-review summarization using fine-tuned Transformer language models / Lucas Roberto da Silva; advisor: Sérgio Colcher. – Rio de janeiro: PUC-Rio , Departamento de Informática, 2021.

v., 64 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática.

Inclui bibliografia

1. Informática – Teses. 2. Sumarização. 3. Ajuste Fino. 4. Modelos de Linguagem. 5. Transformers. I. COLCHER, SÉRGIO. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Acknowledgments

To my adviser Professor Sergio Colcher for the stimulus and partnership to carry out this work.

To CNPq and PUC-Rio, for the aids granted, without which this work does not could have been accomplished.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

PUC-Rio - Certificação Digital Nº 1820996/CA

Abstract

Silva,Lucas Roberto da; COLCHER, SÉRGIO (Advisor). **Unsupervised multi-review summarization using fine-tuned Transformer language models**. Rio de Janeiro, 2021. 64p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Automatic summarization is the task of generating concise, correct, and factual summaries. The task can be applied to different textual styles, including news, academic publications, and product or place reviews. This dissertation addresses the summary of multiple evaluations. This type of application stands out for its unsupervised nature and the need to deal with the redundancy of the information present in the reviews. The automatic summarization works are evaluated using the ROUGE metric, which is based on the comparison of n-grans between the reference text and the generated summary. The lack of supervised data motivated the creation of the MeanSum architecture, which was the first neural network architecture based on an unsupervised model for this task. It is based on auto-encoder and has been extended to other works, but none explored the effects of using the attention mechanism and auxiliary tasks during training. The present work is divided into two parts: the first deals with an experiment in which we make extensions to the MeanSum architecture, adding attention mechanisms and auxiliary sentiment classification tasks. In the same experiment, we explore synthetic data to adapt supervised models for unsupervised tasks. In the second part, we used the results previously obtained to carry out a second study on fine-tuning pre-trained Transformer language models. The use of these models showed a promising alternative to the unsupervised nature of the problem, outperforming previous works by + 4 ROUGE.

Keywords

Summarization; Fine-Tuning; Language Models; Transformers;

PUC-Rio - Certificação Digital Nº 1820996/CA

Resumo

Silva,Lucas Roberto da; COLCHER, SÉRGIO. **Sumarização automática de multiplas avaliações utilizando ajuste fino de modelos de linguagem Transformers**. Rio de Janeiro, 2021. 64p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Sumarização automática é a tarefa de gerar resumos concisos, corretos e com consistência factual. A tarefa pode ser aplicada a diversos estilos textuais, dentre eles notícias, publicações acadêmicas e avaliações de produtos ou lugares. A presente dissertação aborda a sumarização de múltiplas avaliações. Esse tipo de aplicação se destaca por sua natureza não supervisionada e pela necessidade de lidar com a redundância das informações presentes nas avaliações. Os trabalhos de sumarização automática são avaliados utilizando a métrica ROUGE, que se baseia na comparação de n-gramas entre o texto de referência e o resumo gerado. A falta de dados supervisionados motivou a criação da arquitetura *MeanSum*, que foi a primeira arquitetura de rede neural baseada em um modelo não supervisionado para essa tarefa. Ela é baseada em auto-encoder e foi estendida por outros trabalhos, porém nenhum deles apresentou os efeitos do uso do mecanismo de atenção e tarefas auxiliares durante o treinamento do modelo. O presente trabalho é dividido em duas etapas. A primeira trata de um experimento no qual extensões à arquitetura do *MeanSum* foram propostas para acomodar mecanismos de atenção e tarefas auxiliares de classificação de sentimento. Ainda nessa etapa, explora-se o uso de dados sintéticos para adaptar modelos supervisionados a tarefas não supervisionadas. Na segunda etapa, os resultados obtidos anteriormente foram utilizados para realizar um estudo sobre o uso de ajuste fino (*fine-tuning*) de modelos de linguagem *Transformers* pré-treinados. A utilização desses modelos mostrou ser uma alternativa promissora para enfrentar a natureza não supervisionada do problema, apresentando um desempenho de + 4 ROUGE quando comparado a trabalhos anteriores.

Palavras-chave

Sumarização; Ajuste Fino; Modelos de Linguagem; Transformers;

PUC-Rio - Certificação Digital Nº 1820996/CA

Table of contents

PUC-Rio - Certificação Digital Nº 1820996/CA

List of figures

PUC-Rio - Certificação Digital Nº 1820996/CA

List of tables

PUC-Rio - Certificação Digital Nº 1820996/CA

List of symbols

MRS – Multiple Review Summarization

NLP – Natural Language Processing

PUC-Rio - Certificação Digital Nº 1820996/CA

1 Introduction

The internet is an incredible tool for sharing information. Users who buy a product, hire a service, make plans for a trip or for a place to eat, very frequently write their experiences on dedicated websites, personal blogs, forums, or social networks. All of this review data is a valuable resource for building opinion-based applications. One possible application is automatic text summarization, especially *Multiple Review Summarization* (MRS).

Automatic Text summarization is the task of reducing the length of a large piece of text, maintaining most of its relevant information. Review text poses additional specific characteristics: it is opinionated, written by multiple people without a defined style, and can vary on how the text is formally structured (Bražinskas et al., 2020).

The process of text summarization has three key components: the input, the model, and the scoring functions to measure the input element's relevancy.

The input tells how many documents we have to summarize and what features these documents have. Single document summarization is the most popular task addressed in the research field. The popularity of this task is due to its nature. It is a more straightforward problem due to its many annotated datasets. Compared to Multi-Document summarization, the single-document setting, at least in theory, does not suffer from the redundancy problem. Multi-document summarization also suffers from its larger input size, making most of its models challenging to scale. In MRS, multiple reviews are used as the input to generate a concise piece of text containing some representation of the prevalent opinions and main points taken from the various reviews.

PUC-Rio - Certificação Digital Nº 1820996/CA

The second piece is the model, which is where most research efforts are focused. Most models produce their summaries using one of two main approaches: *extractive* or *abstractive* (El-Kassas et al., 2021). Extractive models weigh and select critical pieces of text from the input and use them directly, based on some form of ordering method, to produce the output. On the other hand, abstractive methods generate new text conditioned on the input, maintaining the original sense, but using different phrasing. While extractive models are clearly not as flexible as we could possibly envision for a good summarization technique, abstractive systems allow a degree of freedom that might be regarded as a double-edged sword: while the model is, in theory, capable of generating more "human-like" versions of the input, it can also miss key factual information during its generation step. The default training setup comprises selecting

PUC-Rio - Certificação Digital Nº 1820996/CA

a benchmark dataset and the development of models to transform the text. In the supervised setting, the model can use a set of summaries generated by humans as the target. When these target summaries are not available, we have an unsupervised setting. The release of labeled datasets such as DUC[1] and CNN news (Hermann et al., 2015) created an excellent environment for the development of supervised models. At the same time, the unsupervised task was still struggling to get momentum in the research community. Unlike news articles that use an inverted pyramid structure (Kryscinski et al., 2019) and can have summaries when shared online (Grusky et al., 2018), most of the more general review data does not have a clear structure that makes an automatic construction of reliable labeled datasets something that is easy to produce. Moreover, when dealing with multiple reviews, summarizing models have to comp with conflicting opinions, extract relevant arguments and grounding information, and manage redundancy. These characteristics make unsupervised MRS one of the most challenging among the automatic summarization tasks (Chu and Liu, 2019).

Lastly, to complete the basic description of automatic text summarization, we need to address the scoring system used to evaluate the model's output. ROUGE (Ganesan, 2018) has been the default metric to evaluate summarization systems. It has been heavily criticized because of its lack of a better correlation with human's hand-made summaries. More specifically, its n-gram based formula is incapable of scoring well for abstractive models that change the summary words (Kryscinski et al., 2019). This weakness raises questions about ROUGE's usage as a guiding metric while, at the same time, better alternatives are not widely recognized in the field.

Nevertheless, none of its challenging characteristics stopped the research community from making significant progresses in review summarization in recent years. With the release of new models and the democratization of deep learning frameworks, it has never been easier to get up to speed with state-of-the-art. Cloud providers have increased the offer to GPUs at lower costs, which favored independent researchers who develop their models without relying on an extensive infrastructure that years ago would only be possible in large companies or research institutes. The research on deep learning turned into a very competitive and dynamic environment.

In the Multi-review summarization landscape, the lack of labeled datasets created a good environment for extractive models. They can work based on heuristics or more general optimization techniques (Varalakshmi K and Kallimani, 2018) that model the selection of linguis-

[1]https://duc.nist.gov/duc2004/

tic elements expected to be present in the summary. But, with the rise of deep learning in other fields such as machine translation and headline generation, recent advances in Supervised Summarization have also shown a progressive shift to deep learning as the main framework. Works such as *Get to the point* (See et al., 2017), *bottom-up summarization* (Gehrmann et al., 2018), and pre-trained language models show that Recurrent Neural Networks (RNNs) (Schuster and Paliwal, 1997) and Transformers (Vaswani et al., 2017) can significantly improve ROUGE results over their predecessors. In the same period, within the deep NLP community, fine-tuned pre-trained models dominated many Natural Language Understanding (NLU) benchmarks. Models trained on the masked language task,(Lewis et al., 2020, Raffel et al., 2020, Zhang et al., 2020) for example, showed outstanding results when fine-tuned to perform a downstream task.

This progress arrived to the MRS setting with the release of *MeanSum* (Chu and Liu, 2019), a multi-document unsupervised review summarization system that uses auto-encoder concepts combining a similarity and reconstruction loss to generate its summaries. It was the first end-to-end system to generate abstractive summaries. Many works that followed MeanSum extended its auto-encoder approach, incorporating other strategies to push the unsupervised review summarization benchmark results even further.

This dissertation aims to train neural network models to generate a paragraph-sized summary from multiple reviews, maintaining the overall sentiment and factual consistency. To achieve that, we focus on two approaches. The first approach implements architectural changes to the MeanSum model. In particular, we extend its base architecture by replacing the aggregation function from a mean to attention (Rush et al., 2015) module and include sentiment-based auxiliary tasks. Our second approach, in its turn, adapts some well established News Summarization Models (like (Lewis et al., 2020, Raffel et al., 2020, Zhang et al., 2020, See et al., 2017)) using fine-tuning. This strategy acknowledges the significant progress observed in the supervised news summarization task and explores ways to harness its advantages in an unsupervised setting using synthetic and crowd-sourced data.

We evaluate all models on both benchmarks of the Yelp dataset[2]. All experiments use ROUGE (Ganesan, 2018) as the primary evaluation metric, but Additionally, we also explore the BERTscore (Zhang et al., 2019), and the Coverage and Density (Grusky et al., 2018) as complementary metrics to access the model's performance full picture.

It is worth mentioning that we developed this work when the state of the

[2]https://www.yelp.com/dataset

PUC-Rio - Certificação Digital Nº 1820996/CA

art on neural abstractive multi-review summarization models shifted many times. We adapted our work to match the latest standards. As a result, at the moment of this writing, to the best of our knowledge, we were able to achieve the state of art on this task using pre-trained models and the proposed fine-tuning. Our experiments show how pre-trained Transformer models can achieve the state of the art results after being fine-tuned with less than 100 samples. This result goes against the overall perception that the lack of unsupervised data is the main limitation in the field, showing that only a handful of labeled examples are enough to outperform other models.

This dissertation is structured as follows: firstly, we describe the field of automatic text summarization and the shift to deep learning, describing the characteristics of the sequence-to-sequence framework and unsupervised review summarization. Secondly, we outline the Meansum architecture, the models developed in this work, the transformers used, and our fine-tuning strategy. We then explain the experimental setting, the dataset, and the evaluation metrics. Lastly, we show and discuss how the results relate to the current state of the art and point to possible research directions.

PUC-Rio - Certificação Digital Nº 1820996/CA

2
Literature Review

This work uses techniques from both summarization and language modeling. Therefore, this chapter describes recent advancements on both fields. We start by giving a general view of unsupervised multi-document summarization. We then narrow our review to works directly related to our approach, namely, neural unsupervised multi-review summarization and transformer-based language modeling.

2.1
The Field of Unsupervised Multi-Document Summarization

We used academic search engines to find unsupervised summarization articles published in conferences or journals since 2014. Since our work is focused on neural abstractive summarization, we chose 2014 as the starting point because it was one year before the publication of one of the first successful models (Rush et al., 2015) that used attention with the sequence-to-sequence framework to perform abstractive summarization. Reviewing this period, we can assess how the field developed when attention was released. We made a query[1] for the academic search engines SCOPUS[2] and dblp[3] designed to capture summarization models that do not use supervised learning. Our main question during the review was: 'what is the current state of the field of unsupervised multi-document summarization?'.

Based on our data, we found that most of the work was done on the news domain, and the datasets DUC[4] and TAC[5] were the most widely used by these models. Likewise, the few Models designed for opinion summarization used Amazon (Ni et al., 2019), IMDb[6], Yelp datasets [7] or IMDb [8], the last one being the least popular – with only one use.

Most of these works explored the task using optimization and extractive methods with a similar pipeline, which comprises: (i) the application of

[1](unsupervised OR opinion OR ((semi OR weakly) AND supervised)) AND ((summarization OR summarizing) AND (multi-document OR (multiple AND (reviews OR tweets OR documents))))
[2]https://www.scopus.com
[3]https://dblp.org
[4]https://duc.nist.gov/duc2004/
[5]https://tac.nist.gov/data/past/2011/Summ11.html
[6]https://www.imdb.com/interfaces/
[7]https://www.yelp.com/dataset
[8]https://www.imdb.com/interfaces/

PUC-Rio - Certificação Digital Nº 1820996/CA

preprocessing, such as parts of speech tagging, stop word removal, to the raw text, followed by (ii) a scoring algorithm and (iii) a selection strategy.

Works such as (Huang et al., 2014) defined new heuristics to generate its summaries while (Hong and Nenkova, 2014) extract and combine parts of the sentence using Integer Programming to select the best combination. The key point of change of the optimization approach is the ranking system to select the best sentences.

Works published after 2016 (Chen et al., 2016, Ma et al., 2016) started adopting *word embeddings* (Mikolov et al., 2013) to build dense representations of words . After that year, using dense representations from an embedding has become part of most of the approaches. We found that the availability of datasets such as DUC, TAC, and the more recent Multi-News (Fabbri et al., 2019) was a key factor in the popularity of the supervised approach.

Our search results also showed that systems that are based on neural networks use the sequence-to-sequence setting in the form of an encoder-decoder. The results returned two articles (Lebanoff et al., 2018, Zhang et al., 2018) that propose models based on adapting single document models to work with multiple documents. Both papers use optimization techniques to adjust the attention weights of the underlying model. (Lebanoff et al., 2018) uses Maximal Marginal Relevance (MMR) (Goldstein and Carbonell, 1996) to adjust the attention weights of a pointer-generator network model, while (Zhang et al., 2018) uses a hierarchical encoder (Tan et al., 2017) to compute document representations that are used as context for the decoder and then applies the normalized PageRank (Tan et al., 2017) weights taken from the top K sentences as an attention during the decoding steps. Both argue that the optimization step helps the underlying model to deal with the redundancy problem found in multiple document reviews by only attending to the K most relevant sentences.

Only one paper (Liu and Lapata, 2019) used a Transformer to perform multiple document summarization. More specifically, it proposes a hierarchical transformer that can summarize long documents with the help of an LSTM classifier (HochreiterSepp and SchmidhuberJürgen, 1997) as a content selection step, trained on the ground truth summaries. The selected paragraphs are used as input for the transformer that has local and global layers. The local layers encode paragraphs, while the global layers enable information exchange between paragraphs. They also argue that a graph-based attention mechanism can replace any of the transformer heads in the global layer.

We also searched for works that proposed augmented data as a means

PUC-Rio - Certificação Digital Nº 1820996/CA

to bridge the gap between the supervised and unsupervised model. The only works that explore this king of approach are (Parida and Motlicek, 2019, Nikolov and Hahnloser, 2020). The first article (Parida and Motlicek, 2019) evaluates the model on a German Wiki data [9] using data extracted from Common Crawl [10] as synthetic summaries. They used a reverse model to generate text from the summary, then used the generated text as additional information during training. The second paper (Nikolov and Hahnloser, 2020) created its data using two methods: (i) document and sentence alignment, in which the authors use a hierarchical search algorithm (Nikolov and Hahnloser, 2019) to find pseudo-parallel sentences, and (ii) an LSTM model that generates more data based on the summary sentences.

Evaluation of summarization models has been a recurrent theme of debate. While ROUGE has been widely used as the default metric, (Kryscinski et al., 2019) discusses its shortcomings and argues that the lack of better evaluation metrics may prevent an adequate assessment of the current state of the art on summarization tasks. They claim that, since ROUGE favors lengthy extractive summaries, a better-defined task is a way to improve on its shortcomings.

A new metric, BERTScore (Zhang et al., 2019), based on contextual embeddings, was proposed to address some of the limitations of word-based evaluation metrics. It can be used for many text generation tasks such as image captioning and neural machine translation. It uses the pairwise cosine similarity to compare tokens from each sentence. BERTScore's authors argue that the model has the ability to encode semantic information and thus provides a flexible way to match tokens compared to exact n-gram matches used in ROUGE. Their experiments gives some support that it exhibits better results when compared to other automatic evaluation metrics on machine translation and image captioning tasks.

Two other metrics, *Coverage* and *Density* (Grusky et al., 2018), can be used to measure how extractive the model is. They measure how many words are shared between summary and reference and the average extractive length present on the summary. A *dense model* is highly extractive, while a *sparse model* with high coverage is more abstract. Contrary to previous metrics, Coverage and Density do not aim to rank models; this descriptive approach makes its scores a helper metric that gives more information about the data. It has all the limitations of word-based metrics but, even with the limitations, it delivers a good value on aiding the overall performance assessment.

[9]https://www.swisstext.org/
[10]http://commoncrawl.org/

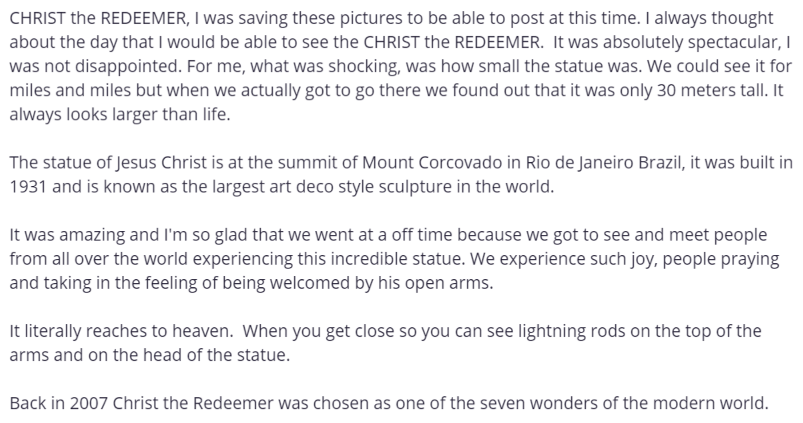
PUC-Rio - Certificação Digital Nº 1820996/CA

2.2 Related works

In the NLP domain, similarly to what was observed in other related areas, modern unsupervised review summarization models follow the trend of deep learning. As mentioned before, MeanSum (Chu and Liu, 2019) was the first neural model to perform unsupervised review summarization. Even though its reign was short-lived, it was able to frame the task as a sequence-to-sequence problem and attract the attention of the NLP community. After its release, four new models followed its developments and extended the base auto-encoder architecture using other types of models. For example, DenoiseSum (Amplayo and Lapata, 2020) and CopyCat (Bražinskas et al., 2020) are based on variational and denoising autoencoders. OpinionDigest (Suhara et al., 2020) and Oposum (Angelidis and Lapata, 2018) are aspect-based models, that include a sentiment phrase extraction step in the pipeline to guide the generation of better summaries. Lastly, FewSum (Bražinskas et al., 2020) uses an extended encoder-decoder structure, and includes a fine-tuning step to adjust the writing style of the generator.

Both DenoiseSum and FewSum explore a different characteristic of the task, taking into account that reviews are different from summaries. They argue that reviews are more personal, describe the user's overall experience, have many personal pronouns. Reviews also describe the day or why the reviewer went to a place and other additional information that cannot be used in a summary. An example of a yelp review can be seen in Fig 2.1.

PUC-Rio - Certificação Digital Nº 1820996/CA



CHRIST the REDEEMER, I was saving these pictures to be able to post at this time. I always thought about the day that I would be able to see the CHRIST the REDEEMER. It was absolutely spectacular, I was not disappointed. For me, what was shocking, was how small the statue was. We could see it for miles and miles but when we actually got to go there we found out that it was only 30 meters tall. It always looks larger than life.

The statue of Jesus Christ is at the summit of Mount Corcovado in Rio de Janeiro Brazil, it was built in 1931 and is known as the largest art deco style sculpture in the world.

It was amazing and I'm so glad that we went at a off time because we got to see and meet people from all over the world experiencing this incredible statue. We experience such joy, people praying and taking in the feeling of being welcomed by his open arms.

It literally reaches to heaven. When you get close so you can see lightning rods on the top of the arms and on the head of the statue.

Back in 2007 Christ the Redeemer was chosen as one of the seven wonders of the modern world.

Figure 2.1: Review of Cristo Redentor. The first and third paragraph have information mixed with personal the personal context of the writer. source: https://www.yelp.com/biz/cristo-redentor-corcovado-rio-de-janeiro

They also claim that it is a challenge to convert all this personal information into a summary. According to their reasoning, a summary should

PUC-Rio - Certificação Digital Nº 1820996/CA

be more descriptive, capturing the many aspects of the object, and give a realistic view of what it offers. This becomes an issue when the training data contains only reviews while the model is asked to generate summaries. In its approach, DenoiseSum discarded reviews that used too many unnecessary symbols. In contrast, FewSum's approach was two-fold: it included a fine-tuning step to their training scheme, and developed a new dataset which is more aligned to the task to provide a more realistic evaluation of the models. These two models hold the highest scores on unsupervised multi-review summarization. DenoiseSum has 30.1 ROUGE on MeanSum's benchmark, and Fewsum has 37.3 ROUGE on its Benchmark dataset.

The last topic explored in our literature review is the use of pre-trained language models in NLP. The use of pre-trained models to ease the training of new models is not a new tactic. Models such as word2vec (Mikolov et al., 2013) have a widespread use for both researchers and practitioners. They replace the bag-of-word approach on most modern NLP problems. More recently, such embeddings were also used to perform extractive summarization (Rossiello et al., 2017). These representations are more powerful than their bag of words counterpart, but still have a significant limitation: its representation is static. After the model is trained, every word has the same vector regardless of the context it appears. This issue motivated the development of *contextual embeddings*. These embeddings deliver representations that are sensitive to context and can adapt the final representation depending on the usage of the word in the text. The contextual representation BERT (Devlin et al., 2019) is one of the most successful contextual representation models. With its release, many text classification benchmarks were beaten just by adding BERT embeddings. Contextual embeddings can also replace static embeddings in the centroid approach to build an extractive summarization model (Miller, 2019).

However, BERT was not designed to perform text generation tasks. Its architecture lacks a decoder element to transform the contextual embeddings into a sequence of text. The models that followed the success achieved by BERT (Lewis et al., 2020, Raffel et al., 2020, Zhang et al., 2020) included this decoder element and extended the Masked Language modeling task to perform text generation. A detailed description of these models is given in chapter 6.1.

With the increase in popularity of large-scale models, much work has been put on the associated discussions about the training of these models, their increasing demand for computational power, their data security risk, and the ethical concerns (Bender et al., 2021, Carlini et al., 2020). Nevertheless, researchers still explore the inner workings of BERT (Rogers et al., 2020) and, even after analyzing 150 papers, it seems that the community does not fully understand how the model learns and uses its linguistic capabilities.

PUC-Rio - Certificação Digital Nº 1820996/CA

3
Baseline models

This chapter gives a detailed description of all models used directly in this work, diving into their inner workings and the significant elements, contributing to their notoriety. We include it to give the appropriate attention to the models's parts that are essential to the development of this dissertation. It gives a complete description of the Transformer (Vaswani et al., 2017) and MeanSum (Chu and Liu, 2019), as they are the base for all other models described herein. For the derived models, such as BERT (Devlin et al., 2019) and Copycat (Bražinskas et al., 2020), we describe the improvements over the base. The chapter is composed of two sections: one describing the models based on MeanSum and the other covering the models based on the Transformer.

3.1
Meansum-related Models

As the pioneer, MeanSum quickly went from state-of-the-art status to a baseline model. The model is praised because it framed the task as a sequence-to-sequence problem, allowing the application of a wide range of deep learning techniques. As table 3.1 depicts, MeanSum outperforms the extractive baseline but is 10 ROUGE behind the current state of the art.

Model	Rouge 1	Rouge 2	Rouge L
FewSum (Bražinskas et al., 2020)	**37.72**	**9.92**	**22.76**
Copycat (Bražinskas et al., 2020)	28.12	5.89	18.32
MeanSum (Chu and Liu, 2019)	27.5	3.54	16.09
TextRank (Pan et al., 2019)	26.96	4.93	16.13

Table 3.1: The table shows the gap between ROUGE scores on MeanSum and FewSum, source: (Bražinskas et al., 2020)

3.1.1
MeanSum

MeanSum proposes an end-to-end LSTM encoder-decoder model that does not need any kind of supervision. Training is possible because of a combination of a similarity loss between the summary and the input representations, and a reconstruction loss (cross-entropy) to predict the next token of the input using its representation. The data flow can be divided in two paths; each loss has its path and data is transformed accordingly.

PUC-Rio - Certificação Digital Nº 1820996/CA

The preprocessing phase uses a *WordPiece* model (Sennrich et al., 2016), which is a tokenization scheme that splits the words into sub-words units that can be combined to reconstruct the original word. With this strategy one can have a small vocabulary of sub-word units (32k) that can represent every word in the vocabulary. The language model is an LSTM with the same size as the encoder module, trained using a cross-entropy loss on the prediction of the next token. After training, this weights are used to initialize the summarization module.

During training, the summarizer encodes the input reviews (X_{enc}), takes the mean of the representations (X_{mean}), and uses it as a context vector for the decoder. The decoder is then trained with two loss functions.The first loss function is a cross-entropy loss, that uses X_{enc} to reconstruct the input using cross-entropy comparing the predicted token with the input token, as depicted in Figure 3.1. For the second loss, the decoder uses X_{mean} to produce

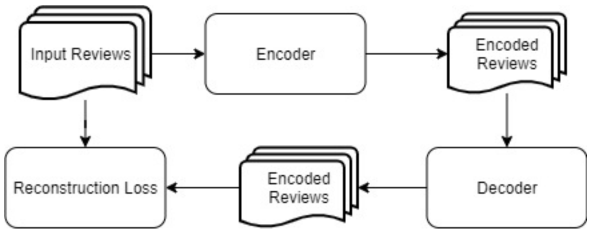


Figure 3.1: Reconstruction loss dataflow

a summary (S) using the Gumbel Softmax (Jang et al., 2016), which can produce differentiable samples from a discrete distribution, and then encode back the produced summary (S_{enc}). Then, it takes the average cosine similarity between S_{enc} and X_{enc} as depicted in Figure 3.2. These losses ensure that the

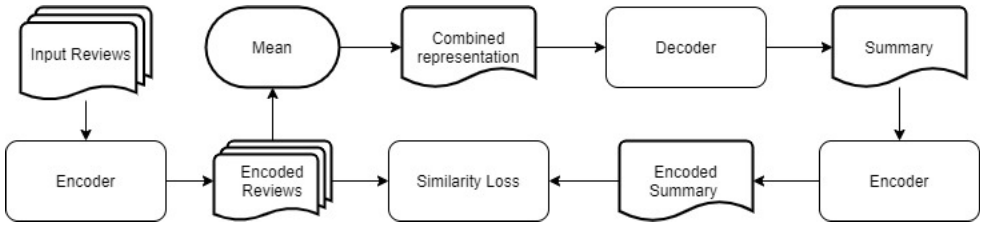


Figure 3.2: Similarity loss dataflow

generated summaries stay inside the domain of the reviews and are similar to the source content. Figure 3.3 shows the complete diagram for the architecture. Figure 3.4 shows an example of a summary generated by MeanSum. It shows that MeanSum included three aspects of the source (service, food quality, and size of the restaurant) while the extractive summary (Rossiello et al., 2017) has very little useful information. We can also note that MeanSum makes a

PUC-Rio - Certificação Digital Nº 1820996/CA

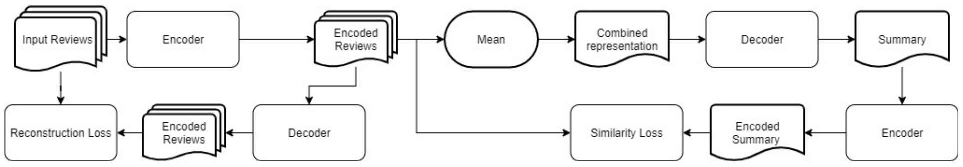
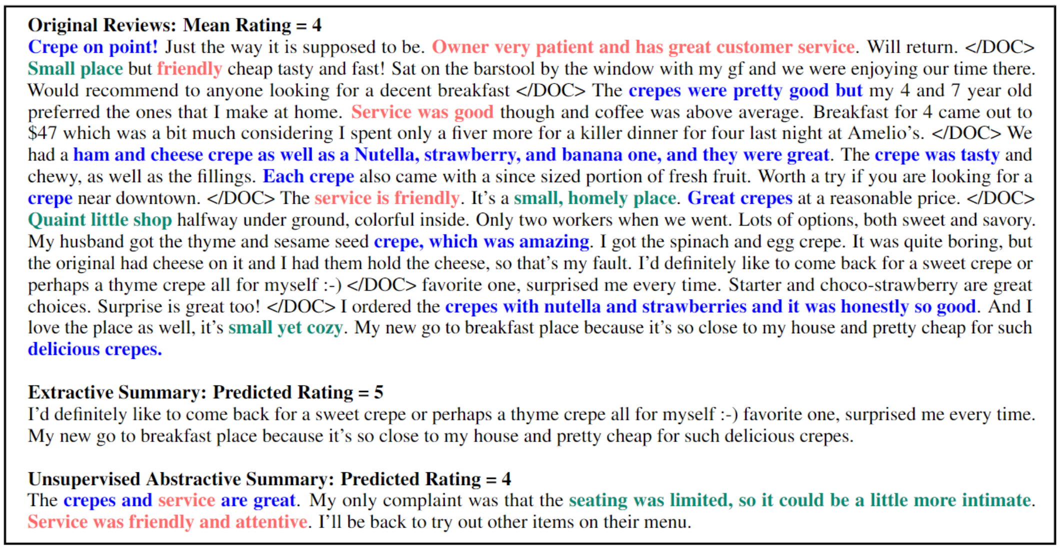


Figure 3.3: MeanSum Complete architecture

mistake when summarizing the fact that the place is small, it generates a sentence without much sense or connection with the rest of the summary.


Original Reviews: Mean Rating = 4
Crepe on point! Just the way it is supposed to be. **Owner very patient and has great customer service**. Will return. </DOC> **Small place** but **friendly** cheap tasty and fast! Sat on the barstool by the window with my gf and we were enjoying our time there. Would recommend to anyone looking for a decent breakfast </DOC> The **crepes were pretty good but** my 4 and 7 year old preferred the ones that I make at home. **Service was good** though and coffee was above average. Breakfast for 4 came out to $47 which was a bit much considering I spent only a fiver more for a killer dinner for four last night at Amelio's. </DOC> We had a **ham and cheese crepe as well as a Nutella, strawberry, and banana one, and they were great**. The **crepe was tasty** and chewy, as well as the fillings. **Each crepe** also came with a since sized portion of fresh fruit. Worth a try if you are looking for a **crepe** near downtown. </DOC> The **service is friendly**. It's a **small, homely place**. **Great crepes** at a reasonable price. </DOC> **Quaint little shop** halfway under ground, colorful inside. Only two workers when we went. Lots of options, both sweet and savory. My husband got the thyme and sesame seed **crepe, which was amazing**. I got the spinach and egg crepe. It was quite boring, but the original had cheese on it and I had them hold the cheese, so that's my fault. I'd definitely like to come back for a sweet crepe or perhaps a thyme crepe all for myself :-) </DOC> favorite one, surprised me every time. Starter and choco-strawberry are great choices. Surprise is great too! </DOC> I ordered the **crepes with nutella and strawberries and it was honestly so good**. And I love the place as well, it's **small yet cozy**. My new go to breakfast place because it's so close to my house and pretty cheap for such **delicious crepes.**

Extractive Summary: Predicted Rating = 5
I'd definitely like to come back for a sweet crepe or perhaps a thyme crepe all for myself :-) favorite one, surprised me every time. My new go to breakfast place because it's so close to my house and pretty cheap for such delicious crepes.

Unsupervised Abstractive Summary: Predicted Rating = 4
The **crepes and service are great**. My only complaint was that the **seating was limited, so it could be a little more intimate**. **Service was friendly and attentive**. I'll be back to try out other items on their menu.

Figure 3.4: Comparison between MeanSum and Centroid; source: (Chu and Liu, 2019)

3.1.2 CopyCat

Copycat (Bražinskas et al., 2020) uses a variational auto-encoder (VAE) based on the text-VAE model(Bowman et al., 2016). It trains an hierarchical representation to encode both reviews and products. The model also uses the pointer-network mechanism (See et al., 2017) to enable access to individual reviews during inference.

3.1.3 OpinionDigest

OpinionDigest (Suhara et al., 2020) is a summarization pipeline (Fig 3.5) that combines an opinion phrase extractor and a Transformer to generate summaries conditioned on reviews, and a set of opinion phrases. The phrases can be used to control which aspects of the reviews the generator should focus. Their complete pipeline has three steps: opinion extraction, selection, and generation. The opinion extractor is a pre-trained model (Miao et al., 2020)

PUC-Rio - Certificação Digital Nº 1820996/CA

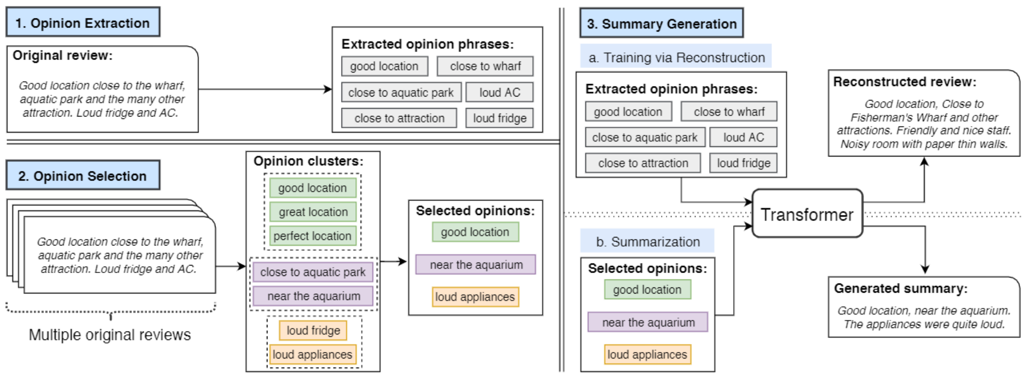
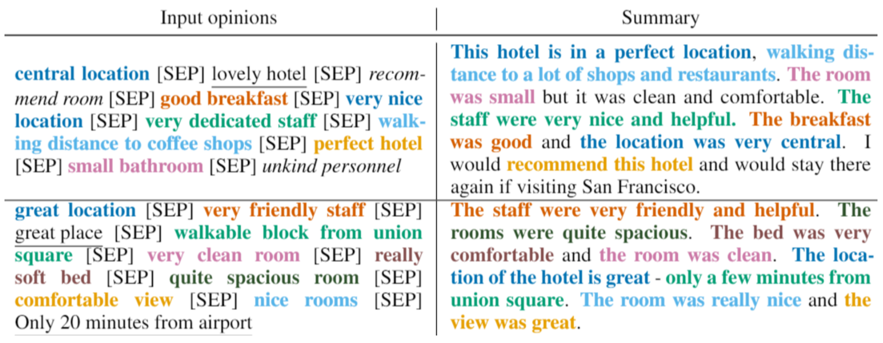


Figure 3.5: OpinionDigest pipeline source: (Suhara et al., 2020)

that outputs opinion phrases, their polarity, and aspect categories. The opinion selection creates opinion clusters based on the cosine similarity of phrases. Clusters only keep novel phrases and discard repeated content. The generation step uses a transformer to reconstruct the input review based on its phrases. Picture 3.6 shows example input/output pairs.



Input opinions	Summary
central location [SEP] lovely hotel [SEP] *recommend room* [SEP] **good breakfast** [SEP] **very nice location** [SEP] **very dedicated staff** [SEP] **walking distance to coffee shops** [SEP] **perfect hotel** [SEP] **small bathroom** [SEP] *unkind personnel*	**This hotel is in a perfect location**, **walking distance to a lot of shops and restaurants**. **The room was small** but it was clean and comfortable. **The staff were very nice and helpful.** **The breakfast was good** and **the location was very central**. I would **recommend this hotel** and would stay there again if visiting San Francisco.
great location [SEP] **very friendly staff** [SEP] great place [SEP] **walkable block from union square** [SEP] **very clean room** [SEP] **really soft bed** [SEP] **quite spacious room** [SEP] **comfortable view** [SEP] **nice rooms** [SEP] Only 20 minutes from airport	**The staff were very friendly and helpful.** **The rooms were quite spacious.** **The bed was very comfortable** and **the room was clean**. **The location of the hotel is great - only a few minutes from union square**. **The room was really nice** and **the view was great**.

Figure 3.6: OpinionDigest input/output example source: (Suhara et al., 2020)

PUC-Rio - Certificação Digital Nº 1820996/CA

3.1.4 DenoiseSum

DenoiseSum (Amplayo and Lapata, 2020) builds a denoise autoencoder trained on a synthetic dataset. The dataset is built using token, chunk, and document level noising functions. All the noising functions replace the content of the input with similar text based on a language model. During training, a sample of the reviews is used as a summary, while its corrupted versions are used as input. The same denoising model is used during generation. Figure 3.7 shows an example of noise variations of the same review.

3.1.5

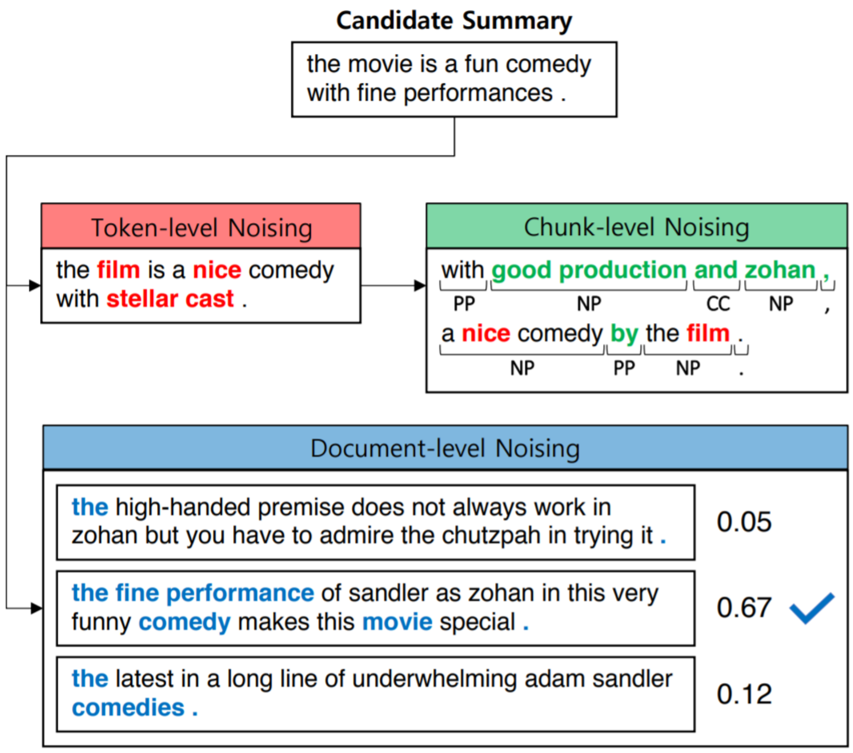

Candidate Summary
the movie is a fun comedy with fine performances .
Token-level Noising
the film is a nice comedy with stellar cast .
Chunk-level Noising
with good production and zohan ,
PP NP CC NP ,
a nice comedy by the film .
NP PP NP .
Document-level Noising
the high-handed premise does not always work in zohan but you have to admire the chutzpah in trying it .
0.05
the fine performance of sandler as zohan in this very funny comedy makes this movie special .
0.67 ✓
the latest in a long line of underwhelming adam sandler comedies .
0.12

Figure 3.7: Noise examples, a candidate summary is used to generate alternative versions changing tokens, chunks and documents based on their similarity to the input. source: (Amplayo and Lapata, 2020)

PUC-Rio - Certificação Digital Nº 1820996/CA

FewSum

FewSum (Bražinskas et al., 2020) employs a transformer and a plug-in feed-forward network to generate summaries. Its transformer is trained using a regularization term that controls the use of words that do not appear in previous reviews. After the unsupervised training, the plug-in network is fine-tuned to adjust the generator model's style to output text that is similar to a summary instead of a review. This paper also introduced a new benchmark dataset. This new dataset's main change is that reviewers were asked to create a more abstract summary with less copied content and a more appropriate text style.

3.2 Transformer Language Models

The Transformer is the base for the large language models that hold the current state of the art in many language generation tasks, including summarization. They are trained on the masked language modeling task introduced by BERT (Devlin et al., 2019) and extended in various ways in other articles [1]. All these models follow the same pre-training then fine-tune paradigm in-

[1]https://huggingface.co/transformers/

troduced by BERT. They also extend SpanBERT (Joshi et al., 2020) on the use of spans instead of single tokens during the pre-training phase. BART and T5 were released around the same time and are both inspired by SpanBERT. While they both try to find a model that solves many tasks, Pegasus is focused only on abstractive summarization.Table shows Rouge results for the CNN Daily Mail dataset (Hermann et al., 2015) 3.2, we can see that pegasus is the best model by a very small margin.

	R1	R2	RL
T5	43.52	21.55	40.69
BART	44.16	21.28	40.9
Pegasus	**44.17**	**21.47**	**41.11**

Table 3.2: Transformer results on CNN Daily Mail dataset

3.2.1 The Transformer Architecture

The transformer architecture (Vaswani et al., 2017) was proposed as an alternative for LSTMs as the base model used in sequence-to-sequence tasks. They argue that the attention mechanism is enough to retain long range dependencies in sequences and that the sequential nature of recurrent models make its computational requirements too high for long inputs.The architecture is built using a multi-head attention block followed by a position wise feed forward neural network that can be trained parallely. Since it does not rely on convolutions or recurrence it uses a positional encoding to give the model relative information about its input position in the sequence. Figure 3.8 show a general views of the model and Figure 3.9 shows its Multi-Head attention and Point-wise Feed forward network.

3.2.1.1 Multi-Head attention

After the positional encoding, the input X is projected by a set of projection matrices to form the Query,Key and Value vectors used by the attention block.

$$Q = XW_q \tag{3-1}$$

$$K = XW_k \tag{3-2}$$

$$V = XW_v \tag{3-3}$$

PUC-Rio - Certificação Digital Nº 1820996/CA

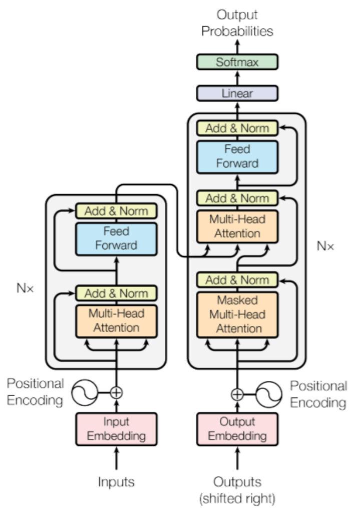


Figure 3.8: Overview of the Transformer architecture; source: (Vaswani et al., 2017)

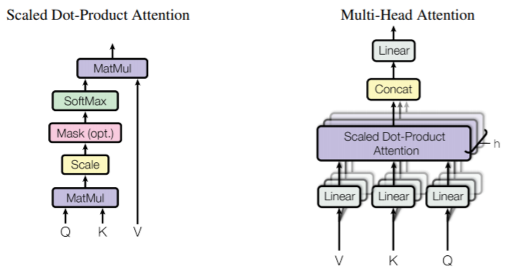


Figure 3.9: Transformer main modules; source: (Vaswani et al., 2017)

where Wq, Wk, Wv are the learnable projection matrices. The attention weights are computed using a scaled version of the dot attention (Luong et al., 2015).

$$Attention(head) = \frac{(QK^T)}{\sqrt{d_k}}V \tag{3-4}$$

The model computes this attention function parallely, the number of attention heads is a hyperparameter and $d_k = outputdimension/numberofheads$, they used 8 heads and the output dimension of 512 in the paper. The multi-head attention is the concatenation of each attention head followed by a linear layer

$$MultiHeadAttention = Concat(h_0, ..., h_n)W_o \tag{3-5}$$

The output of the MultiHead Attention block is

$$LayerNorm(x + MHA(x)) \tag{3-6}$$

Where MHA is the MultiHead Attention layer and LayerNormis a layer normalization described at (Ba et al., 2016).

PUC-Rio - Certificação Digital Nº 1820996/CA

Objective	Inputs	Targets
Masked Language modeling	the cat jumped [MASK] the lazy dog	over
Next Sentence Prediction	[CLS] the cat jumped over [SEP] the lazy dog [SEP]	isNextSentence

Table 3.3: BERT uses the special tokens [MASK], [SEP] and [CLS]. [MASK] and [SEP] are used during pre-training and [CLS] is used for sentence classification taks.

3.2.1.2 Position-Wise Feed Forward Neural Network

This layer is a linear transformation of its input. It employs two fully connected feed forward networks as linear transformation.

$$PWFFN(x) = L_2(Relu(L_1(x))) \tag{3-7}$$

Where L_1, and L_2 are linear transformations. The output of the position-wise is also fed too a residual connection followed by a layer normalization.

$$LayerNorm(x + PWFFN(x)) \tag{3-8}$$

The self-attention between encoder layers use the input or the output previous layer. Between encoder and decoder, the outputs of the encoder are used for the Key and Value, and the decoder output is used for the Query. Lastly, the inputs for the decoder are masked during training to emulate sequential training, the transformer does not have sequential training so the masking prevents information leakage. The transformer is the base model for state of the art language models like BART and Pegasus, those models showed the ability to model long range dependencies keeping the correct grammatical structure and factual consistency when generating text unlike most abstractive summarization systems.

PUC-Rio - Certificação Digital Nº 1820996/CA

3.2.2 BERT

BERT (Devlin et al., 2019) encoder only transformer model pre-trained on masked language modeling and next sentence prediction that achieved estate of the art GLUE (Wang et al., 2018) results when released. It introduced the pre-train then finetune framework, were a model is trained on large text dataset on linguistic tasks, that may not be aligned to the final task, then finetuned on the target task. It is trained on the masked language modeling task, where 15% of the input is masked and the model has to correclty predict the masked tokens, and next sentence prediction, which is a binary classification problem that given two sentences it classifies if they occur in sequence or not, figure 3.3 shows the task description.

The relationship between the pre-training task and the ability to learn linguistic features is still a open question.It is undeniable that bert-like models have amazing performance on benchmark problems but it does not translates to a better understanding of the underling language (Rogers et al., 2020).

3.2.3 BART

BART (Lewis et al., 2020) sequence to sequence model based on BERT and GPT (Radford et al., 2019). It extends the masked language modeling pre-training objective by including four new tasks for the model to solve, Figure 3.10, and a general framework to train such models. This framework's key point is using a function that adds noise to the input, corrupting the sequence the model has to reconstruct. The article describes four new corruption strategies and compares their use with the masked token prediction seen in BERT. The combination of text infilling, sentence permutation, and masked language modeling achieved up to 3.5 ROUGE in gains on text generation tasks. The new tasks are:

- **Sentence Permutation**: The pre-processor splits the input into sentences and randomly change its order.
- **Document Rotation**: A random token is selected to be the new start of the document. The tokens before it are moved to the end of the input.
- **Token Deletion**: A ratio of the input is randomly selected to be deleted. The model has to replace the missing tokens with the correct one.
- **Text Infiling**: Masks a sample of continuous tokens are selected to be masked; the model has to predict the correct number of masked tokens it has to reconstruct from only one mask.

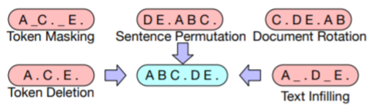


Figure 3.10: BART taks visual descritpion; source: (Lewis et al., 2020)

PUC-Rio - Certificação Digital Nº 1820996/CA

PUC-Rio - Certificação Digital Nº 1820996/CA

3.2.4 T5

T5 (Raffel et al., 2020) uses a Text-to-Text framework where the model is trained to solve many tasks at once. They accomplish this by introducing a new training scheme using prefix conditioning. In this setting, the model is pre-trained as a masked language model and then trained on multiple tasks. Each task has a prefix, and the model is fine-tuned to distinguish between all tasks. Surprisingly, the prefix's choice did not influence the final performance of the model. They used 'summarization:' for text summarization, 'translate <a> to <b>:', as Figure 3.11 shows, for machine translation between languages a and b.They also introduce a batch sampling scheme that balances multiple tasks between batches.

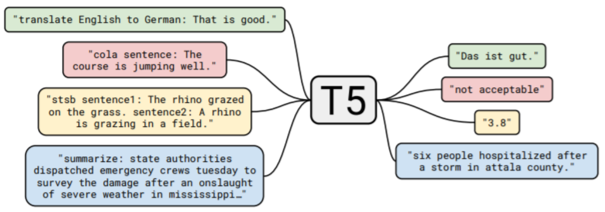

"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

Figure 3.11: Description of an input for varying tasks. T5 can learn how to perform based on the prefix and expected output.; source: (Raffel et al., 2020)

3.2.5 Pegasus

Pegasus (Zhang et al., 2020) introduces the Gap Sentences Generation that masks entire sentences instead of small text span. They also evaluate if sentence choice impacts performance and present experiments with three selection schemes. The first selection strategy selects the first M sentences, the second randomly selects M sentences, and the last one uses top M sentences ranked by Rouge. The results showed that selecting leading sentences worked best for news datasets but worse on other domains. The author suggests that choosing the sentences based on Rogue works best for downstream tasks.

4 Proposed Models

This chapter describes all models developed in this work. It starts by outlining our first experiment and what models were developed, namely: the attention, hierarchical attention, auxiliary sentiment tasks, and training of supervised models using augmented data. Then, we describe the fine-tuning experiment using Transformer models.

4.1 First Experiment: MeanSum Extensions and Augmented Data

The MeanSum uses an LSTM encoder-decoder that takes multiple reviews as input to produce an abstractive summary. The model merges these representations fusing the mean, generating the context vector for the decoder. We develop two approaches to improve on MeanSum's Results: a training scheme that uses both attention, as shown in Fig 4.1, and auxiliary tasks use of attention instead of the mean; and the use of augmented data as a bridge between supervised and unsupervised models.

Attention has shown great results when used in news summarization, and we wanted to reproduce this improvement. For our second approach, since the training data does not have paired summaries, we can use extractive models to generate a summary and use it as the target for a supervised model. The use of augmented data allows the training of state-of-the-art supervised models for this unsupervised task. We train a pointer-generator (See et al., 2017) from scratch, and fine-tune BART (Lewis et al., 2020), T5 (Raffel et al., 2020), Pegasus (Zhang et al., 2020) using this strategy.

In more detail, this first experiment replaces the pure LSTM decoder of the Meansum model with an attentive decoder. It also includes two sentiment objectives to aid the model during training: a sentiment classification model (that uses the encoder input to classify the reviews), and a sentiment regressor that predicts the mean sentiment of the summaries. The literature review did not show any model that combines multiple objectives during training in the same way we are doing.

In a second study still within this experiment, we included state-of-the-art supervised summarization models using generated data to adapt established news summarization models to the multi-review summarization context. We hypothesize that the extractive models can select critical information, and the neural model can join all sentences in a human-like way. By combining

PUC-Rio - Certificação Digital Nº 1820996/CA

both models, we intend to have factual consistency from extractive methods and, at the same time, achieve the fluidity of neural models.

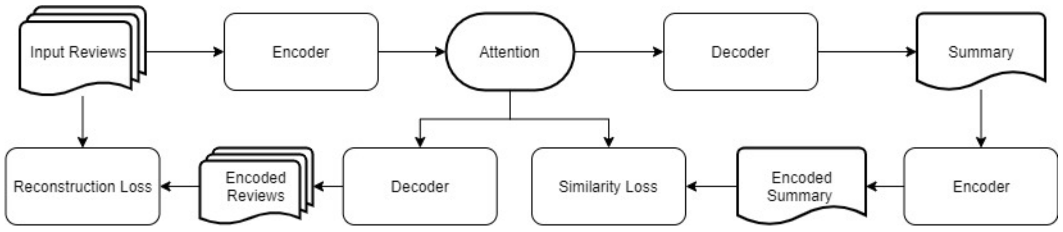


Figure 4.1: Proposed model

4.1.1 Additive Attention and Pointer-Generator Network

In a sequence-to-sequence setting using the encoder-decoder architecture, the input is a source sequence X, with length n, and the target is a sequence Y. Using a bidirectional LSTM as encoder and a LSTM as decoder, the hidden states of the encoder are denoted as

$$\begin{aligned}\overrightarrow{h} &= \overrightarrow{enc}(X)\\ \overleftarrow{h} &= \overleftarrow{enc}(X)\\ h_i &= [h_i; h_i]\\ c &= f(hi)\end{aligned}$$

where c is the context vector computed using some reduction function over the hidden states (in MeanSum the reduction function is the mean of the hidden states). The decoding step for a timestamp t of Y uses the last hidden state of the decoder, the context vector, and the last generated item of Y

$$y_t, s_t = dec(y_{t-1}, s_{t-1}, c) \tag{4-1}$$

where s_t represents the hidden state of the decoder at timestamp t. This architecture has a limitation that c has to keep all the information about the source sentence. The decoder has to use this information during all its decoding steps, making it difficult to generate long sentences.

The attention mechanism (Bahdanau et al., 2014) was proposed to ease the decoder job when decoding long sequences by letting the decoder attend to different parts of the input each step. With attention, the model can learn a differentiable function of the encoder's hidden states that weighs the importance of each hidden step.

$$a_t = softmax(\bar{h}) \tag{4-2}$$

PUC-Rio - Certificação Digital Nº 1820996/CA

where $\bar{h}$ will be computed using the function

$$\bar{h} = v_a^T tanh(W_a s_t + W_b h_i) \ additive \ attention \tag{4-3}$$

where W_a, W_b are larnable weights. With attention, each decoding step has its own context vector, the vector can be computed using the following equation:

$$c_t = \sum_{i=0}^{i=t-1} a_t^i h_t^i \tag{4-4}$$

Where i denotes the index of the hidden state for each input. This work also includes the pointer-generator network model proposed by (See et al., 2017). Let P_{gen} be the probability of generating a word from the vocabulary.

$$P_{gen} = sigmoid(W_h \bar{h} + W_s s_t + W_x x_t) \tag{4-5}$$

where W_h, W_s and W_x are learnable weights. The next word is generated using P_{gen} to weight the attention and vocabulary distributions,

$$P(w) = P_{gen} * P_{vocab} + (1 - P_{gen}) * a_t \tag{4-6}$$

where P_{vocab} is the distribution over the words of the vocabulary, it can be computed by two linear layers after concatenating the attention vector with to the decoder hidden state, $P_{vocab} = softmax(W_b(W_a[s_t; \bar{h}]))$ where W_a, W_b are learnable parameters of the linear layers. The model is trained using negative log likelihood loss over $P(w)$. Since this loss needs a target signal we use it for the reconstruction step of MeanSum replacing the cross entropy loss. Another contribution from (See et al., 2017) is the coverage mechanism, and it prevents the model from attending to the same word repeatedly. Let c_t be the coverage vetor, it is defined as the cumulative attention from all decoding step,

$$c_t = \sum_{i=0}^{i=t-1} a_i h_i \tag{4-7}$$

The coverage vector is also included in the attention calculation

$$\bar{h} = v^T tanh(W_h h_i + W_s s_t + W_c c_t) \tag{4-8}$$

To penalize the model when it attends to the same word they included the coverage loss

$$covloss = \sum_{i=0}^{i=t-1} c_i \tag{4-9}$$

Where i is the index of an element of the sequence. In this setup the final loss function becomes

$$loss = -log(P(w^*)) + \sum_{i=0}^{i=t} a_i c_i + lsim \tag{4-10}$$

PUC-Rio - Certificação Digital Nº 1820996/CA

where $lsim$ is the similarity loss.

4.1.2 Hierarchical Attention

The input for the model has the following shape:

(**B**ath size, number of **D**ocuments, max sequence **L**ength, model **dim**ention).

This shape follows the same pattern present in hierarchical models (Yang et al., 2016). An attentive hierarchical model can better handle the input size and learn how to attend in two levels, as shown in (Yang et al., 2016), the hierarchical model learns weights for both sentences and words. Since our work deals with multiple documents, it uses both document and token attention. The final model still has the same overall architecture. The hierarchical attention is used as a two-step reduction function, as described in the equation below, by first computing attention weights for document representations before attending to the input at a token level. This attention is implemented using an additive attention model to encode both token and document level information. Given an input with the shape (B, D, L, dim), as described in the previous paragraph.

$$\begin{aligned} &tokencontext = Attention(input)\ \#output\ shape\ (B, D, dim) \\ &docmentcontext = Attention(tokencontext)\ \#output\ shape\ (B, dim) \end{aligned} \tag{4-11}$$

PUC-Rio - Certificação Digital Nº 1820996/CA

4.1.3 Sentiment and Regression models

Review summarization may be differentiated from news summarization by the presence of a sentiment element that is not present in news context (Chu and Liu, 2019). Therefore, a model that aims to summarize reviews should capture and express this sentiment. To align our model to the task's written description, we hypothesize if a sentiment model's inclusion during training would improve the final performance.

Research on sentiment classification developed many models to solve the task. The Yelp review dataset is one of the benchmarks for the task. It has two main approaches. The first is Yelp polarity, which replaces lower star reviews with negative sentiment and high star reviews with positive sentiment. The other is Yelp fine-grained, which tries to classify the correct number of starts a review has. The fine-grained problem is much more challenging than its two labels counterpart. Some of the challenges are a class imbalance problem and too much ambiguity between middle-range reviews. Table 4.2 shows the development of the polarity task; even simple models have very high accuracy while the fine-grained task, Table 4.1, is still a challenge to the state of the art transformer models.

model name	**acc**
BERT Large	**70.68**
ULMFiT	70.02
DPCNN	69.42
DRNN	69.15
BERT large finetune UDA	67.92

Table 4.1: Top models Yelp finegrained on Paper With Codes.

model	**acc**
BERT large	**98.11**
BERT large finetune UDA	97.95
ULMFiT	97.84
DPCNN	97.36
DRNN	97.27

Table 4.2: Top models polarity on Paper With Codes.

PUC-Rio - Certificação Digital Nº 1820996/CA

The challenge to include another model during the training is that the sentiment model has to be good enough to classify sentiment on its own. This way, it avoids the case where the model is too weak to identify the correct sentiment when used with MeanSum. Simultaneously, it cannot be a large model and take all the computation time during training. We started with a simple architecture with an LSTM followed by only fully connected layers, but it could not get more than 60% accuracy. We then include an attention layer but were still not enough to learn from our sample of the dataset. Our final model uses a batch normalization layer (Ioffe and Szegedy, 2015) before the final projection layer, feeding normalized data to the final projection layer. The normalized output helps to smooth the optimization problem leading to faster and better convergence.Table 4.3 shows the results of our classification models.In contrast to an encoder-decoder setting, classification problems do not require a decoder. It uses an auxiliary vector to serve as context. This vector mimics the decoder state, but it is trained together with the model as a variable. The regression model has the same architecture and takes the same input as the similarity loss. It uses the generated summary to predict the reviews' mean sentiment, outputting a single real value from its last projection layer.

model	accuracy	fscore	loss
lstm_att_256_256_norm	**0.716**	**0.647**	0.685
lstm_att_128_128_norm	0.713	0.634	**0.670**
lstm_att_256_512	0.709	0.641	0.681
transformer_128	0.686	0.597	0.733
transformer_512	0.675	0.578	0.758
lstm_att_128_256	0.645	0.539	0.845
lstm_128_256	0.522	0.356	1.177

Table 4.3: The LSTM with attention is better than the basic transformer but worse than its batch norm equivalent. The small model with batch norm has better accuracy than a bigger attention only model.

4.2 Second Experiment: Fine-tuning Transformers

The fine-tuning process (Goodfellow et al., 2016) follows a standard pipeline . This setup allows the model to leverage its linguistic power on the first pass, then adjust the whole style and semantic knowledge when fully trained. Training the whole model without training the last layer can lead to worse results or divergence (Rogers et al., 2020).

Fine-Tuning Pipeline

1. Load pre-trained model with frozen weights
2. Initialize the last layer with random weights
3. Train the model
4. Unfreeze the whole model
5. Train the model

We adopt this textbook approach to further our point that fine-tuning should be considered when developing new summarization models. We also include a domain transfer exploration to evaluate how fine-tuned models generalize when evaluated out of domain. An extensive overview (Rogers et al., 2020) of what has been learned about BERT (Devlin et al., 2019) showed that there is no best way to fine-tune these models. Every fine-tuning strategy has its pros and cons, and a deep exploration of which fine-tuning strategy is best for this task is beyond the scope of this work. The fine-tuning can yield comparable results with less compute and answer the field's main limitation: The lack of paired summaries.

PUC-Rio - Certificação Digital Nº 1820996/CA

5
Methodology

This Chapter describes how we structured our data pipeline, hardware settings, datasets, and model configurations for our experiments. We use google Colab [1] to train all models in a sample of the Yelp dataset using the same optimizer settings for all models. The experiments focus on the model features, and thus the ablation studies only change the size of the model, varying the number and size of each layer.

5.1
Development details

The project uses Allennlp [2] as the primary deep learning framework. It is a PyTorch-based deep learning framework designed for NLP research, making it easy to experiment with new models. It also takes care of the training loop, regularization, and stability methods, such as gradient clipping/normalization and label smoothing (Szegedy et al., 2016). All the models based on MeanSum were trained using the same scheme as the paper. First, we train the model as a language model, predicting the next token for the sequence. We introduce our auxiliary objectives, the similarity, sentiment classification, and mean sentiment regression. Picture 5.1 shows the training path for the model.

PUC-Rio - Certificação Digital Nº 1820996/CA

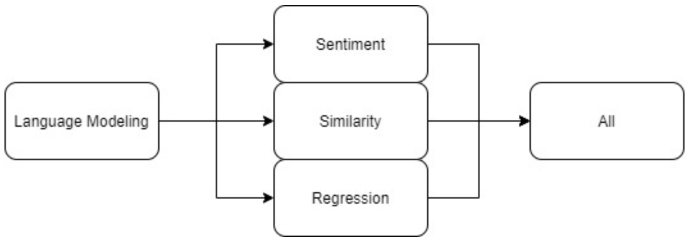


Figure 5.1: Proposed model training pipeline

The main challenge during development was handling large GPU memory usage. The multi-document nature of the problem results in long input sequences. The size of the model is not big, but the operations it requires, together with the nature of the input, makes it hard to train without a large GPU memory.

Colab is an excellent tool for deep learning researchers. It provides a jupyter-like notebook with access to a Tesla V100 GPU with 16GB VRAM and 24GB of RAM. Due to its RAM memory size, we had to sample our

[1]colab.research.google.com
[2]https://github.com/allenai/allennlp

dataset from 2,336,424 to 458,480 reviews. This way, we could fit all data in memory, leading to faster training time and the possibility of batching inputs based on sequence length. We use Evaltools (Fabbri et al., 2020) to evaluate all models with a diverse set of metrics. This tool collects multiple metrics used in NLP in an easy-to-use package. Offering ROUGE, BERTScore, Coverage, Compression, and Density. For the sentiment classification, we use Huggingface Transformers [3] pre-trained model and spacy [4] for general text processing. The same hardware setup is used in The second experiment. The main difference is the use of Blurr[5], a fine-tuning framework that combines the functionality of FastAi[6] with the model library from Huggingface Transformers. Blurr and AllenNLP result from the current trend of AI democratization, where new tools are being developed to ease the access to estate of the art models and best software engineering practices.

5.2 Evaluation

The evaluation process follows past literature and evaluates all models using ROUGE as the primary metric, and BERTScore (Zhang et al., 2019) as auxiliary evaluation. To get the complete picture of the results we compute Compression, Coverage and Density (Grusky et al., 2018). This set of statistics compares the amount and length of extractive segments in the summary to deliver three metrics that can convey how extractive the model is. None of these metrics give a complete view of the model, and there is a need for better evaluation schemes than N-Gran-based metrics (Kryscinski et al., 2019). Unfortunately, even the model based metrics like BERTscore also fail to give low scores to sentences when the semantic meaning is changed, but the syntax remains similar. For example, Comparing 'I love this place' with 'I hate this place' gives 0.97 BERTScore.

5.2.1 ROUGE

The ROUGE-N formulation for a set of reference sentences R and generated summaries S is defined by:

[3]https://huggingface.co/transformers/
[4]https://spacy.io
[5]https://github.com/ohmeow/blurr
[6]https://www.fast.ai

PUC-Rio - Certificação Digital Nº 1820996/CA

$$\begin{aligned}
Precision &= \frac{\#match(S,R)}{\#S} \\
Recall &\frac{\#match(S,R)}{\#R} \\
F1 &= 2\frac{Precision * Recall}{Precision + Recall}
\end{aligned} \tag{5-1}$$

Where $match()$ is the number of overlapping words between the summary and reference. ROUGE-L/W has similar definition but uses the longest common substring instead:

$$\begin{aligned}
Precision &= \frac{LCS(S,R)}{\#S} \\
Recall &= \frac{LCS(S,R)}{\#R}
\end{aligned} \tag{5-2}$$

$\#S$ is the size of the summary and $\#R$ is the size of the reference.

5.2.2 BERTscore

BERTscore (Zhang et al., 2019) uses BERT (Devlin et al., 2019) to generate a representation of the input text that is used for comparison. The language model takes care of encoding the semantic features from the text, and it can compare source and reference using a pairwise cosine similarity between the words. It uses a greedy strategy to compute the score for each word as the maximum similarity between itself and all words from the reference. The final metric is the sum of all scores normalized by the norm of the reference vector. The following formula gives a formal definition of BERTscore: Given a set of token representations y_i of a reference summary Y and x_i from a generated summary X, the score is computed as:

$$\begin{aligned}
R_{bert} &= \frac{1}{|X|}\sum_{x_i \in X} \max_{y_i \in Y} cosine(x_i, y_i) \\
P_{bert} &= \frac{1}{|Y|}\sum_{y_i \in Y} \max_{x_i \in X} cosine(x_i, y_i) \\
F_{bert} &= 2\frac{P_{bert} * R_{bert}}{P_{bert} + R_{bert}}
\end{aligned} \tag{5-3}$$

5.2.3

PUC-Rio - Certificação Digital Nº 1820996/CA

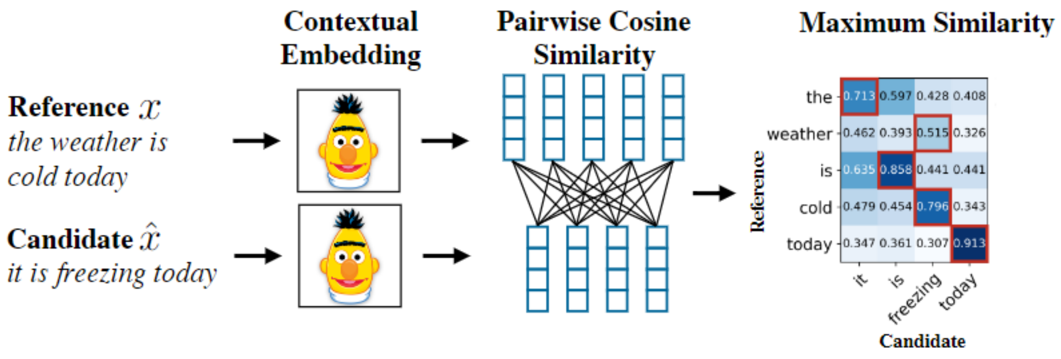


Figure 5.2: Visual representation of BERTscore's computation; source: (Zhang et al., 2019)

Coverage, Compression and Density

Newsroom (Grusky et al., 2018) is a news summarization dataset build using metadata on articles pages. They scrape 38 news sources and collected 1,3 million article-summary pairs from diverse sources and a wide range of summary types present on the article page's HTML tag. The content scraped can be human-made or automatically generated. They propose these three metrics to analyze their vast dataset. A high-density model is highly extractive, and the balance between coverage and density marks a more abstract summary. We choose to include these metrics over other model-based because of their clarity. Model-based metrics are helpful for ranking but not suitable for analysis. When comparing two models using BERTscores, we can only compare how similar they are to the source, but when we combine it with Coverage and Density, we can say which model is more abstractive or cover the mode of the input. Since we are yet to find the one metric to evaluate all, we argue that the combination of multiple metrics is the better way to build a complete view of the models. The formal definition uses the concept of $Fragment$ $F(A, S)$, a shared span between source A and summary S, which is the set of extractive phrases shared between the article and its summary.

5.2.3.1
Coverage

Coverage is the ratio of words shared between source and summary present in the summary.

$$Coverage(A, S) = \frac{1}{|S|} \sum_{f \in F(A,S)} |f| \tag{5-4}$$

5.2.3.2

PUC-Rio - Certificação Digital Nº 1820996/CA

PUC-Rio - Certificação Digital Nº 1820996/CA

Density

Density is the average length of the shared sequences between source and summary in the summary.

$$Density(A, S) = \frac{1}{|S|} \sum_{f \in F(A,S)} |f|^2 \tag{5-5}$$

5.2.3.3 Compression

Compression is the size ratio between the input and the summary.

$$Compression(A, S) = \frac{|A|}{|S|} \tag{5-6}$$

5.2.4 Sentiment Classification

We use a BERT model fine-tuned on the Yelp dataset, with an accuracy of 72%, as our sentiment classifier. The sentiment from the human-generated summary is used as a target and compared to our summary. We also compute the sentiment for every review in our test set and use it to get the mean sentiment. These two measures will help us make sense of how well our models are capturing the main tone of the review together with all other auxiliary metrics.

5.3 Training Data

We use the Yelp dataset for our experiments. It has data about businesses, restaurants, bars, and hotels, including metadata about the attributes of the locations. However, we do not use the metadata from the reviews since most of them are too sparse to use without designing a model around it. The first experiment uses the same filtering criteria described by MeanSum (Chu and Liu, 2019). First, we filter out all reviews with more than 250 tokens, and then we remove all businesses/products with less than 50 reviews. The last step is to split the data into training, validation, and testing following the 80/10/10 ratio. We randomly sampled 458,480 reviews from the training set and 285,448 reviews from the validation set to fit our memory constraints.

Our first experiment also extends the data using TextRank, and every input group is summarized using the extractive model to generate the target sequence. This summary is used during the training of copy-attention and fine-tuning of the language models.

For the second experiment, we use both benchmark datasets released by FewSum (Bražinskas et al., 2020), and MeanSum (Chu and Liu, 2019) during the fine-tuning of our language models. The MeanSum dataset is the same used to benchmark the first experiment, and the FewSum dataset is the new, more abstractive dataset. The new dataset was built to address the writing style issue discussed in both FewSum and DenoiseSum. Both datasets were created similarly, but FewSum's included restrictions to guarantee that summarizers would produce a summary with less copying from source and with a more appropriated writing style. These datasets had defined splits that we followed to maintain comparability. MenSum uses a 100/100 split for training/evaluation, and FewSum defines their 90/90/90 split for training, validation, and evaluation, respectively.

PUC-Rio - Certificação Digital Nº 1820996/CA

PUC-Rio - Certificação Digital Nº 1820996/CA

6
Results

This Chapter describes the results from our experiments, each having a dedicated section to present its results. The last section discuss all scores and their relation with the current works.

We use the base MeanSum performance (model id 20) as the no-change scenario: when models are just as good as the base architecture. Most of our tables have a 'diff' column that shows the Rouge difference between the model and MeanSum. The base MeanSum model follows the architecture described in Section 3.1.1 and all models listed as LM represent language models.

Table 6.1 shows that using augmented data with supervised models gives better results than modeling the task in a completely unsupervised way, with a 6.5 average rouge improvement over the base model. Even the copy-generator (See et al., 2017) model, with about 10% of the transformers' parameters, got comparable scores using augmented data. When improving the base architecture, the attention and the use of auxiliary objectives had a lesser impact on the final performance. Still, they all manage to get better results than the base model, as shown in Picture 6.1. Likewise, Tables 6.8 and 6.9 show that the use of pre-trained transformers achieves better results than its specialized counterparts when fine-tuned using less than 100 human-generated samples.

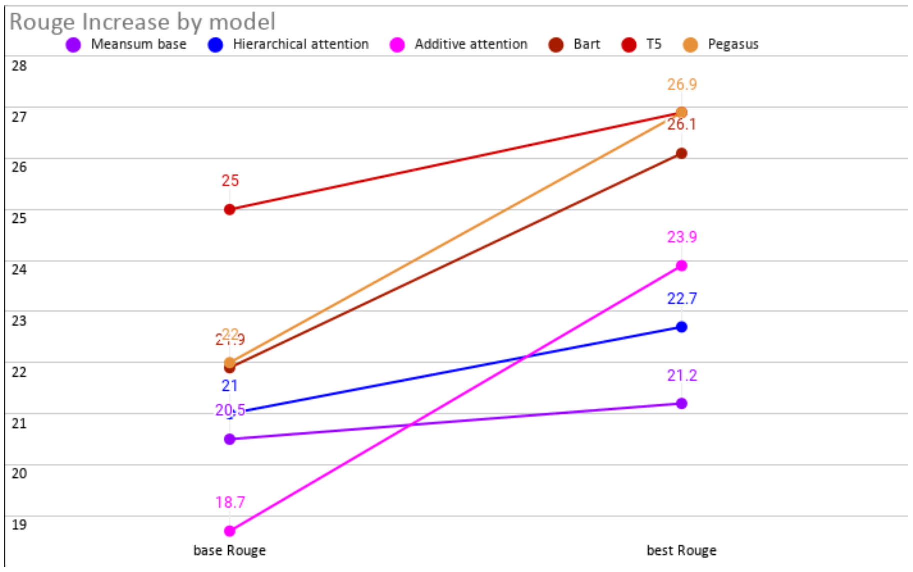


Figure 6.1: The chart summarizes the first experiment results, showing that all models improved when using our methods.

id	model	reg	sim	sent	rouge1	diff	bert_score	coverage	density	accuracy	summary_length
1	**T5**				**26.9**	**6.8**	**0.855**	0.424	0.558	0.49	98.75
2	**pegasus**				**26.9**	**6.8**	0.852	0.365	0.449	0.465	96.16
3	copy generator				26.7	6.6	0.846	0.357	0.442	0.495	170.665
4	bart				26.1	6	0.846	0.362	0.446	**0.52**	**186.065**
6	additive attention		x	x	23.9	3.8	0.827	0.386	0.458	0.37	63.99
7	*hierarchical attention			x	22.7	2.6	0.839	0.456	0.567	0.435	43.8
8	*meansum	x	x		22.7	2.6	0.854	**0.517**	**0.731**	0.385	25.955
20	meansum base		x		20.1	0	0.843	0.51	0.688	0.475	27.135

Table 6.1: Best models from each category. Models trained on augmented data dominate the ranks followed by attention and auxiliary objectives.

6.1 First Experiment

This section shows the results of our first experiment grouping models based on how they are trained. We include a section to discuss the impact of the results and explore the similarity of generated summaries to evaluate different architectures generate similar summaries.

6.1.1 Models Based on Meansum

The overall result, shown in Table 6.2, is that both sentiment and regression achieve better rogue than the base. While the regression objective works well in theory, our data showed that it is not as helpful as its sentiment counterpart. The best way to build the model is to include additive attention and the base model's sentiment objective.Picture 6.2 shows the comparison grouped by model type.

As expected, the language models performed worse than most models. Only when using hierarchical attention that a language model performed better than the base MeanSum. All the attention-based models have a positive average Rouge difference from the base. With the hierarchical attention language models having the best mean, additive attention the best max, and additive attention language models worse performance overall.

Picture 6.3 shows the modes grouped by the number of encoder layers. The models with one encoder layer got the best max performance while two layers show more consistent scores. Most models that performed worse than the base have only one encoder layer. They were trained early on the project to show how model size impacts performance. Our first hypothesis was that bigger models achieve better results, but table 6.3 shows that simpler models can also be competitive with larger ones. Our final chart, Picture 6.4 shows the impact of the auxiliary objectives. As stated before, sentiment classification is the only objective with positive mean and max results on its own. Combining the two tasks also yields positive results, but all additional tasks do not improve over

PUC-Rio - Certificação Digital Nº 1820996/CA

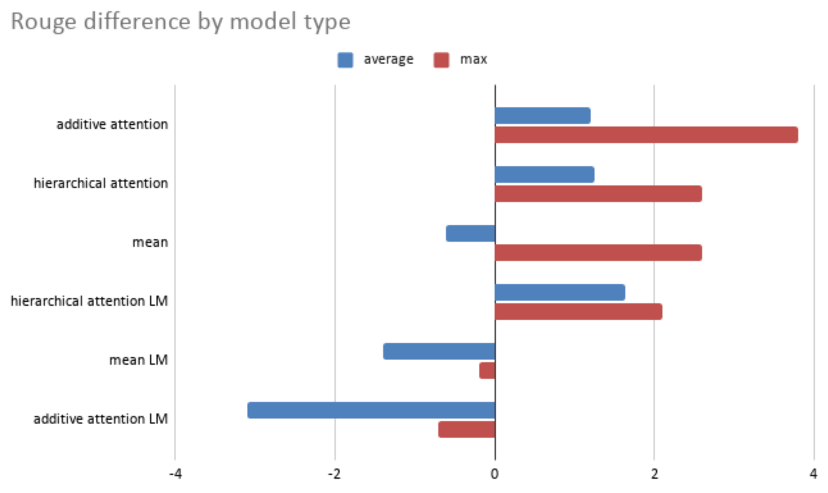


Figure 6.2: Rouge difference from base MeanSum

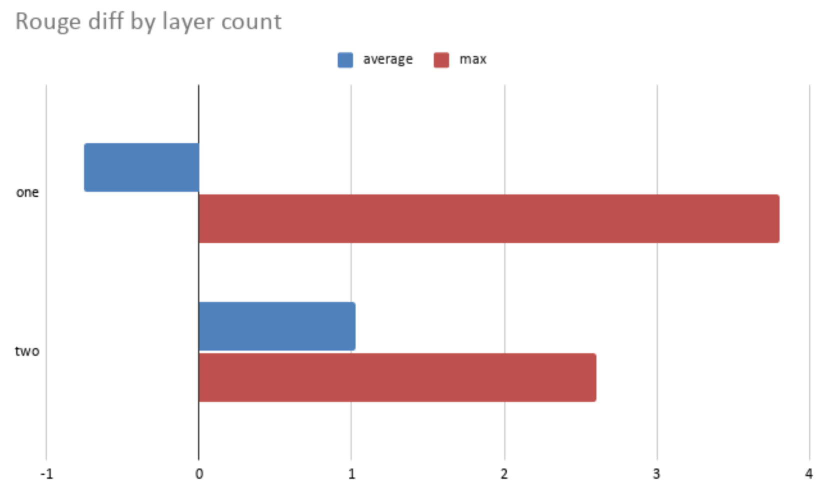


Figure 6.3: Rouge difference split by number of encoder layers

PUC-Rio - Certificação Digital Nº 1820996/CA

the base model. Our initial hypothesis was that all objectives trained together would restrain the model, making it generate better summaries than the base model. However, our results showed that there is no gain in using all goals at the same time.

These results are not a final answer for the impact of all auxiliary tasks. Since we use a simple training pipeline, sequentially training the language model then the summarization model with the combinations of the different objectives. Other combinations and schemes could show better results integrating all tasks. As far as we know, there is no standard way to measure each objective's interaction and influence on the final model. We hypothesize that a more complex training pipeline that uses the theory from multi-task learning can deliver better results when combining the tasks. The search for such a training scheme is beyond this work scope, and it is left to future studies.

PUC-Rio - Certificação Digital Nº 1820996/CA

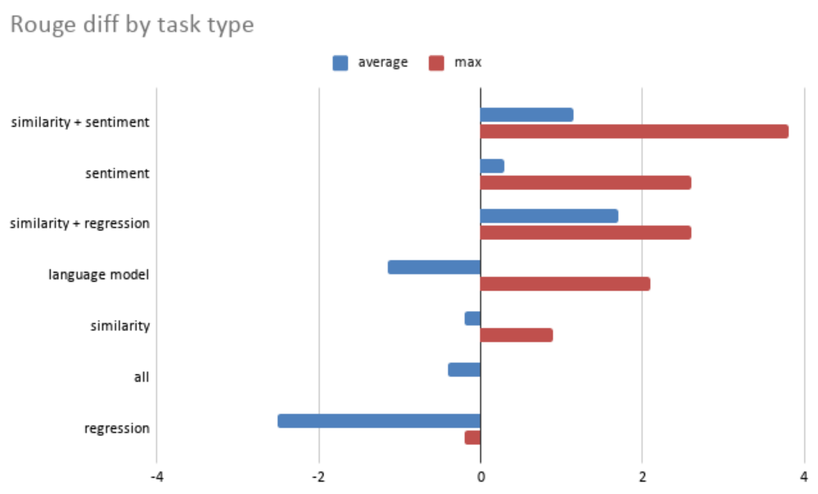


Figure 6.4: Rouge difference grouped by auxiliary task

id	model type base	emb size	hidden size	reg	sim	sent	rouge1	diff	bert_score	coverage	density	accuracy	summary_length
6	**additive attention**	256	256		x	x	**23.9**	3.8	0.827	0.386	0.458	0.37	**63.99**
7	*hierarchical attention	128	256			x	22.7	2.6	0.839	0.456	0.567	0.435	43.8
8	mean	256	512	x	x		22.7	2.6	**0.854**	0.517	0.731	0.385	25.955
9	*hierarchical attention	128	128				22.2	2.1	0.84	0.471	0.589	0.455	37.71
34	*hierarchical attention	128	256	x	x		22.2	2.1	0.836	0.491	0.603	0.46	38.79
35	*hierarchical attention	128	256		x	x	22.2	2.1	0.836	0.491	0.603	0.46	38.79
36	hierarchical attention	128	256		x	x	21.8	1.7	0.84	0.42	0.513	0.45	40.155
12	hierarchical attention	128	256				21.7	1.6	0.839	0.433	0.529	0.455	36.245
14	hierarchical attention	128	256			x	21.6	1.5	0.841	0.424	0.526	0.46	36.745
15	*hierarchical attention	256	512				21.3	1.2	0.839	0.442	0.553	0.4	39.425
16	*mean	256	512		x	x	21.2	1.1	0.836	0.456	0.596	0.44	40.205
17	hierarchical attention	128	256		x		21	0.9	0.841	0.43	0.531	0.425	34.53
18	*mean	256	512	x	x		20.5	0.4	0.835	0.498	0.674	0.475	32.775
19	mean	256	512	x	x	x	20.1	0	0.843	0.533	0.728	**0.495**	28.065
20	mean base	256	512		x		20.1	0	0.843	0.51	0.688	0.475	27.135
21	hierarchical attention	128	256	x			19.9	-0.2	0.838	0.431	0.528	0.44	31.89
22	mean	256	512				19.9	-0.2	0.836	0.522	0.712	0.45	28.97
23	*mean	256	512		x		19.8	-0.3	0.838	0.535	0.716	0.425	28.555
24	additive attention	128	256				19.4	-0.7	0.799	0.559	0.634	0.4	58.75
37	hierarchical attention	128	256	x	x		19.4	-0.7	0.828	0.452	0.545	0.425	37.31
25	*mean	256	512	x	x	x	19.3	-0.8	0.845	0.454	0.623	0.425	24.755
26	additive attention	128	256				19.2	-0.9	0.821	0.415	0.474	0.455	35.94
27	mean	256	512		x	x	19	-1.1	0.836	0.524	0.675	0.465	28.655
28	additive attention	128	256		x		18.7	-1.4	0.796	0.355	0.422	0.36	42.44
29	mean	256	512				17.5	-2.6	0.822	0.585	0.737	0.415	27.82
30	additive attention	128	256				17.3	-2.8	0.822	0.394	0.459	0.365	26.095
31	mean	256	512			x	16.9	-3.2	0.807	**0.672**	**0.806**	0.46	35.96
32	mean	256	512	x			15.3	-4.8	0.812	0.438	0.55	0.41	53.215
33	additive attention	128	256				12.1	-8	0.822	0.372	0.45	0.345	18.95

Table 6.2: All models based on MeanSum. The star marks models with two encoder layers.

6.1.2 Models Trained on TextRank Summaries

Our second group, from our first experiment, has the models trained with augmented data. The copy-generator attention performed surprisingly well, achieving better results than all un-tuned transformers and BART fine-tuned. Copy-generator can learn when to generate or copy words from the input. It was unexpected for it to perform better than the out-of-the-box transformer models, and even a fine-tuned BART, only having around 10% of the number parameters. T5 and Pegasus got the same result, 26.9 ROUGE, only 0.2 more ROUGE than copy-generator. These two are the largest models in our experiment, and it was expected of them to perform well. T5 base got the highest density and coverage, indicating that it is highly extractive, while BART fine-tuned got the best accuracy.

The use of augmented data can be a promising way to leverage unsupervised data. Our data showed that the use of model-based data augmentation is limited. Still, more contemporary research suggests that linguistically motivated heuristics may perform better, as shown in DenoiseSum (Amplayo and Lapata, 2020), where they trained its model on synthetic summaries with great success. The evaluation of augmentation schemes is also an interesting future direction.

model	**attention type**	**rouge1**	**bert_score**	**coverage**	**density**	**accuracy**	**summary_length**
T5 sample	self-attention	**26.9**	**0.855**	0.424	0.558	0.49	98.75
Pegasus sample	self-attention	**26.9**	0.852	0.365	0.449	0.465	96.16
copy-generator	copy-generator	26.7	0.846	0.357	0.442	0.495	170.665
Bart sample	self-attention	26.1	0.846	0.362	0.446	**0.52**	186.065
T5 base	self-attention	25	0.849	**0.446**	**0.575**	0.45	70.93
Pegasus base	self-attention	22	0.851	0.376	0.468	0.4	42.605
Bart base	self-attention	21.9	**0.855**	0.412	0.531	0.375	32.32
Transformer base	self-attention	21.7	0.838	0.362	0.445	0.405	45.49

Table 6.3: Transformer results.

PUC-Rio - Certificação Digital Nº 1820996/CA

Extending this exploration, we evaluate how well transformer models can capture style by fine-tuning them with a small sample of labeled data. Contrary to our augmented data, human-made summaries have all the elements we want our models to capture. BART got the best results in all sample sizes. Pegasus was designed for summarization and got worse results than T5. Table 6.4 shows that Rouge scales linearly with the sample size, increasing for T5 and Pegasus and decreasing for BART. T5 was the most extractive model when trained with 50 and 100 samples. BART got the highest Coverage and Density on ten samples. Pegasus was the most consistent model when conveying the same sentiment as the target summary, achieving the best accuracy on all samples.

Sample	**model**	**N params**	**rouge1**	**coverage**	**density**	**accuracy**	**summary_length**
100	bart	406M	**31.6**	0.448	0.631	0.45	62.27
	T5	770M	**31.6**	**0.498**	**0.717**	0.43	76.41
	pegasus	568M	31.2	0.474	0.675	**0.49**	130.05
50	bart	406M	**32.3**	0.447	0.654	**0.507**	67.6
	T5	770M	30.4	**0.49**	**0.703**	0.433	76.887
	pegasus	568M	29.1	0.417	0.554	**0.507**	104.24
10	bart	406M	**32.5**	**0.472**	**0.68**	0.474	65.495
	T5	770M	27.5	**0.472**	0.658	0.453	77.274
	pegasus	568M	25.3	0.376	0.477	**0.489**	65.858

Table 6.4: Transformer results grouped by Sample size.

6.1.3
First Experiment Discussion

Table 6.7 show the results of all models evaluated on MeanSum benchmark. The Transformers achieved the best results, followed by copy-generator attention. The additive and hierarchical attention were not enough to beat transformer models, but they were better than the base MeanSum almost every time. The same table shows that BERTScore ranks two models with the same score as the best. These two models have a 5 ROUGE gap between them. This result indicates that this metric does not have any easy way to interpret the score, and its contextual representations are not enough to distinguish between positive and negative sentences, leading to very high scores for very distinct sentences.

When we look at the coverage and density, the MeanSum base combined with sentiment classification (model id 31) is the most extractive model. Its short length helps to get high scores. By definition, these metrics are computed using sentence length, and contrary to ROUGE, they favor brief summaries. With this length, shared fragments represent a more significant portion of the total summary resulting in a higher score. The accuracy reveals that our models are not enough to capture the source's overall sentiment, with the most accurate model having around 45% accuracy.

When we evaluate summary similarity, we see that models generate summaries that are similar to their group. We select only models that have more than .6 mean similarity between summaries. Similar architectures generate the most similar summaries, even with a varying range of rouge scores, and using WordPiece (Wu et al., 2016), the models use the same set of words to compose their summaries. The biggest surprise was that the base Transformer has the highest mean similarity with the large transformer models, suggesting that even with later models' improvements, the underlying architecture still chooses similar words to compose the summaries.

Our fine-tuning experiment showed that only the Transformers trained on human summaries compare to current models. A fine-tuned BART achieve 32.5 ROUGE with ten summaries and T5 scores 30.4 with 50 summaries. It shows how valuable the human-generated dataset is. These models do not fail to deliver outstanding results in text generation tasks and show a promising research direction.

6.2

PUC-Rio - Certificação Digital Nº 1820996/CA

source_id	**target_id**	**similarity**	**source_type**	**target_type**
3	4	0.784	augmented data	augmented data
14	17	0.755	meansum variation	meansum variation
1	1	0.752	augmented data	augmented data
17	36	0.751	meansum variation	meansum variation
12	17	0.742	meansum variation	meansum variation
14	36	0.738	meansum variation	meansum variation
12	14	0.734	meansum variation	meansum variation
12	36	0.730	meansum variation	meansum variation
21	17	0.685	meansum variation	meansum variation
24	13	0.683	augmented data	transformer
24	13	0.683	augmented data	transformer
21	36	0.671	meansum variation	meansum variation
12	21	0.666	meansum variation	meansum variation
7	15	0.657	meansum variation	meansum variation
21	14	0.653	meansum variation	meansum variation
28	17	0.636	meansum variation	meansum variation
4	2	0.633	augmented data	augmented data
9	34	0.632	meansum variation	meansum variation
9	35	0.632	meansum variation	meansum variation
17	27	0.629	meansum variation	meansum variation
3	2	0.627	augmented data	augmented data
9	15	0.620	meansum variation	meansum variation
23	25	0.617	meansum variation	meansum variation
4	1	0.616	augmented data	augmented data
3	1	0.613	augmented data	augmented data
28	27	0.610	meansum variation	meansum variation
10	2	0.609	augmented data	augmented data
15	34	0.607	meansum variation	meansum variation
15	35	0.607	meansum variation	meansum variation
9	7	0.604	meansum variation	meansum variation
7	34	0.603	meansum variation	meansum variation
7	35	0.603	meansum variation	meansum variation

Table 6.5: Similarities between the summaries generated by each model.

source_type	**target_type**	**similarity**
augmented data	augmented data	0.662
augmented data	transformer	0.683
meansum variation	meansum variation	0.660

Table 6.6: Mean similarity between models type.

PUC-Rio - Certificação Digital Nº 1820996/CA

id	model	attention type	emb size	hidden size	reg	sim	sent	rouge1	diff	bert_score	coverage	density	accuracy	summary_length
1	T5 sample	self-attention						**26.9**	6.8	**0.855**	0.424	0.558	0.49	98.75
2	Pegasus sample	self-attention						**26.9**	6.8	0.852	0.365	0.449	0.465	96.16
3	opennmt-py-Bottom-Up	copy-generator						26.7	6.6	0.846	0.357	0.442	0.495	170.665
4	Bart sample	self-attention						26.1	6	0.846	0.362	0.446	**0.52**	186.065
5	T5 base	self-attention						25	4.9	0.849	0.446	0.575	0.45	70.93
6	MeanSum	additive attention	256	256		x	x	23.9	3.8	0.827	0.386	0.458	0.37	63.99
7	*MeamSum	hierarchical attention	128	256			x	22.7	2.6	0.839	0.456	0.567	0.435	43.8
8	MeanSum	mean	256	512	x	x		22.7	2.6	0.854	0.517	0.731	0.385	25.955
9	*MeamSum	hierarchical attention	128	128				22.2	2.1	0.84	0.471	0.589	0.455	37.71
34	*MeanSum	hierarchical attention	128	256	x	x		22.2	2.1	0.836	0.491	0.603	0.46	38.79
35	*MeanSum	hierarchical attention	128	256		x	x	22.2	2.1	0.836	0.491	0.603	0.46	38.79
10	Pegasus base	self-attention						22	1.9	0.851	0.376	0.468	0.4	42.605
11	Bart base	self-attention						21.9	1.8	**0.855**	0.412	0.531	0.375	32.32
36	MeanSum	hierarchical attention	128	256		x	x	21.8	1.7	0.84	0.42	0.513	0.45	40.155
12	MeanSum	hierarchical attention	128	256				21.7	1.6	0.839	0.433	0.529	0.455	36.245
13	Transformer base	self-attention	512					21.7	1.6	0.838	0.362	0.445	0.405	45.49
14	MeanSum	hierarchical attention	128	256			x	21.6	1.5	0.841	0.424	0.526	0.46	36.745
15	**MeamSum	hierarchical attention	256	512				21.3	1.2	0.839	0.442	0.553	0.4	39.425
16	*MeamSum	mean	256	512		x	x	21.2	1.1	0.836	0.456	0.596	0.44	40.205
17	MeanSum	hierarchical attention	128	256		x		21	0.9	0.841	0.43	0.531	0.425	34.53
18	*MeamSum	mean	256	512	x	x		20.5	0.4	0.835	0.498	0.674	0.475	32.775
19	MeanSum	mean full	256	512	x	x	x	20.1	0	0.843	0.533	0.728	0.495	28.065
20	MeanSum	mean base	256	512		x		20.1	0	0.843	0.51	0.688	0.475	27.135
21	MeanSum	hierarchical attention	128	256	x			19.9	-0.2	0.838	0.431	0.528	0.44	31.89
22	MeanSum	mean	256	512				19.9	-0.2	0.836	0.522	0.712	0.45	28.97
23	*MeamSum	mean	256	512		x		19.8	-0.3	0.838	0.535	0.716	0.425	28.555
24	lstm beam	additive attention	128	256				19.4	-0.7	0.799	0.559	0.634	0.4	58.75
37	MeanSum	hierarchical attention	128	256	x	x		19.4	-0.7	0.828	0.452	0.545	0.425	37.31
25	*MeamSum	mean	256	512	x	x	x	19.3	-0.8	0.845	0.454	0.623	0.425	24.755
26	MeanSum	additive attention	128	256				19.2	-0.9	0.821	0.415	0.474	0.455	35.94
27	MeanSum	mean	256	512		x	x	19	-1.1	0.836	0.524	0.675	0.465	28.655
28	MeanSum	additive attention	128	256		x		18.7	-1.4	0.796	0.355	0.422	0.36	42.44
29	MeanSum	mean	256	512				17.5	-2.6	0.822	0.585	0.737	0.415	27.82
30	MeanSum	additive attention	128	256				17.3	-2.8	0.822	0.394	0.459	0.365	26.095
31	MeanSum	mean	256	512			x	16.9	-3.2	0.807	**0.672**	**0.806**	0.46	35.96
32	MeanSum	mean	256	512	x			15.3	-4.8	0.812	0.438	0.55	0.41	53.215
33	LSTM greddy	additive attention	128	256				12.1	-8	0.822	0.372	0.45	0.345	18.95

Table 6.7: Compilation of all results.

PUC-Rio - Certificação Digital Nº 1820996/CA

Second Experiment

We explore Transformer language models' capability to learn a new style by fine-tuning, evaluating the models on current Yelp benchmark datasets, and comparing our previous models' scores. Table 6.8 shows the scores for Yelp dataset on FewSum's benchmark and Table 6.9 reports the evaluation on MeanSum's data.

The fine-tuned models achieved better results in both settings using less computational resources. The current trend on Transformer language models is to release new models to be used by the community, enabling this fine-tuning approach to evolve with the general advance of NLP.

Our results show that BART has the highest Rouge in both datasets achieving new SOTA results, followed by FewSum on its benchmark, and T5 and Pegasus when evaluated on MeanSum data.

6.2.1 FewSum Benchmark

Our model achieves comparable results to the state-of-the-art on Yelp dataset using only 90 training examples. Figure 6.5 compares summaries from FewSum and BART. From our models, BART achieved better results in all metrics while T5 got better RL and R2.

6.2.2

model	fine tuning	rouge1	rouge2	rougeL
BART	full	**37.5**	**11.6**	**23.5**
FewSum		**37.3**	09.9	22.8
T5	full	36.1	10.3	**23.1**
Pegasus	full	35.1	09.4	22
T5	last	29.2	06.9	17.6
Copycat		28.1	05.9	18.3
MeanSum		28	03.5	16.1
Pegasus	last	25.8	04.4	16.4
BART	last	25.1	04.7	16.3

Table 6.8: Rouge results on Yelp dataset.

MeanSum benchmark

When evaluated on MeanSum's data, all Transformers achieved better results than previous models. These results show that we can still improve our fine-tuning schemes. This evaluation favors models trained only using the last layer despite the whole model's higher generalization capacity.

model	fine tuning	rouge1	rouge2	rougeL
Bart	last	**34.2**	**07.6**	**19.7**
Pegasus	last	33.9	**07.4**	**19.5**
Pegasus	full	33.7	**07.2**	**19.2**
Bart	full	31.1	05.6	17.9
DenoiseSum		30.1	05.0	17.7
OpinionDigest		29.3	05.7	18.5
T5	full	27.6	04.3	15.8
T5	last	27	03.4	14.8

Table 6.9: Rouge results on MeanSum's benchmark dataset.

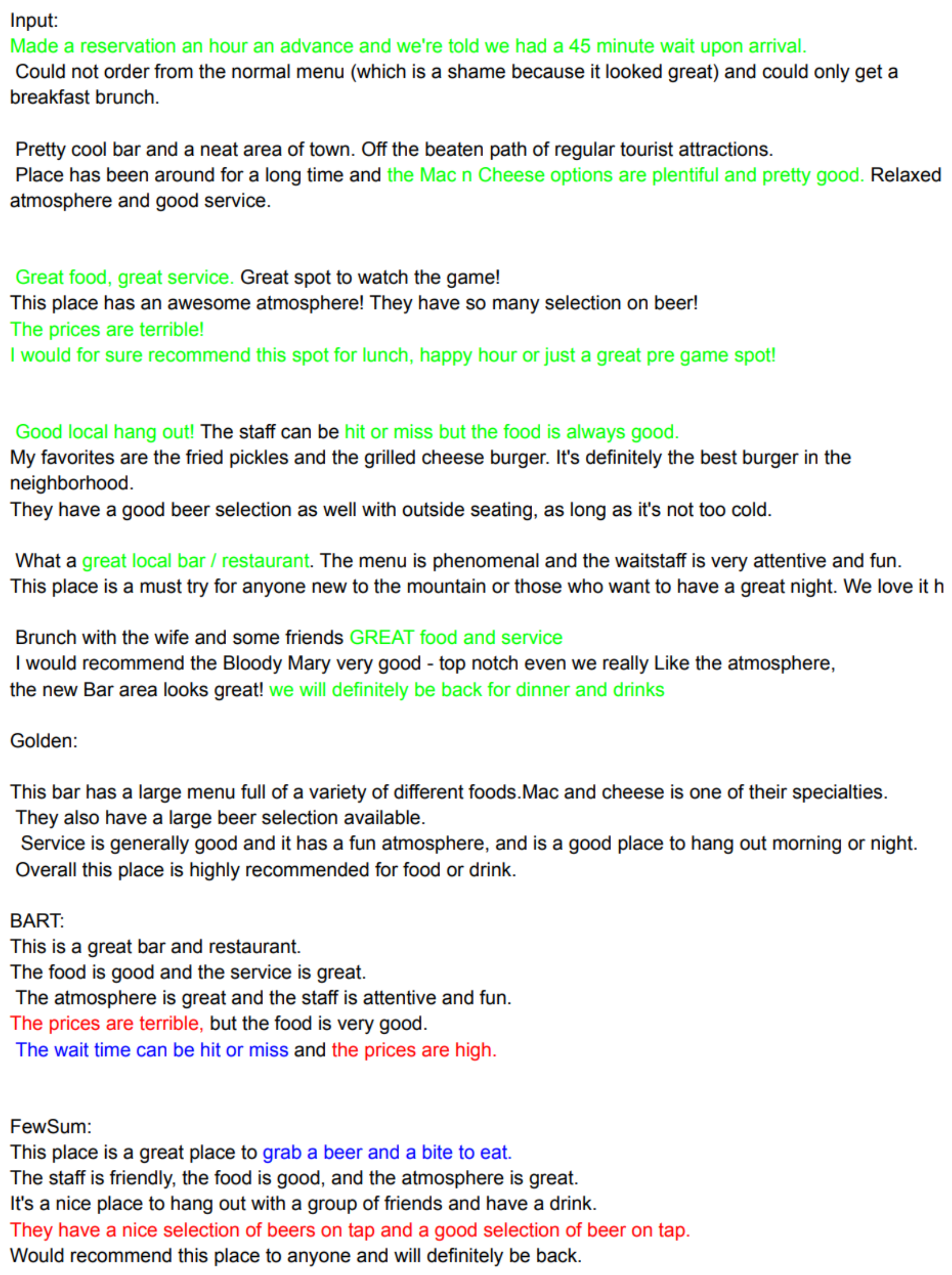
PUC-Rio - Certificação Digital Nº 1820996/CA

6.2.3 Domain transfer

Another lens we can look at the performance of transformer models is to evaluate how they perform out-of-domain.

We perform two experiments fine-tuning the model on the MeanSum data to evaluate on FewSum dataset, and vice versa. Tables 6.10 and 6.11 showed the results on FewSum and MeanSum evaluation data when the model was fine-tuned only using the other dataset and a combined version of the training data.

The overall result is that more data does not mean more performance. All models performed worse than their in-domain variant, and only BART got higher results than previous models.



Input:
Made a reservation an hour an advance and we're told we had a 45 minute wait upon arrival.
Could not order from the normal menu (which is a shame because it looked great) and could only get a breakfast brunch.

Pretty cool bar and a neat area of town. Off the beaten path of regular tourist attractions.
Place has been around for a long time and the Mac n Cheese options are plentiful and pretty good. Relaxed atmosphere and good service.

Great food, great service. Great spot to watch the game!
This place has an awesome atmosphere! They have so many selection on beer!
The prices are terrible!
I would for sure recommend this spot for lunch, happy hour or just a great pre game spot!

Good local hang out! The staff can be hit or miss but the food is always good.
My favorites are the fried pickles and the grilled cheese burger. It's definitely the best burger in the neighborhood.
They have a good beer selection as well with outside seating, as long as it's not too cold.

What a great local bar / restaurant. The menu is phenomenal and the waitstaff is very attentive and fun.
This place is a must try for anyone new to the mountain or those who want to have a great night. We love it h

Brunch with the wife and some friends GREAT food and service
I would recommend the Bloody Mary very good - top notch even we really Like the atmosphere,
the new Bar area looks great! we will definitely be back for dinner and drinks

Golden:

This bar has a large menu full of a variety of different foods.Mac and cheese is one of their specialties.
They also have a large beer selection available.
Service is generally good and it has a fun atmosphere, and is a good place to hang out morning or night.
Overall this place is highly recommended for food or drink.

BART:
This is a great bar and restaurant.
The food is good and the service is great.
The atmosphere is great and the staff is attentive and fun.
The prices are terrible, but the food is very good.
The wait time can be hit or miss and the prices are high.

FewSum:
This place is a great place to grab a beer and a bite to eat.
The staff is friendly, the food is good, and the atmosphere is great.
It's a nice place to hang out with a group of friends and have a drink.
They have a nice selection of beers on tap and a good selection of beer on tap.
Would recommend this place to anyone and will definitely be back.

Figure 6.5: The passages highlighted in green show what pieces from the input are present in the summary. We can see that most of the input is repeated or personal information. The red text show mistakes generated by the models. BART mentioned the price twice and FewSum generated the same sentence, just changed the positive adjective used to describe the beer. The blue text shows the use of casual expressions in the correct context. We can see that both models are able to generate comparable text to the golden summary, but BART's mistake does not make the text less believable.

PUC-Rio - Certificação Digital Nº 1820996/CA

dataset	**model**	**rouge1**	**rouge2**	**rougeL**
combined	Pegasus	**35.9**	**09.2**	21.8
	T5	**35.8**	08.9	**22**
	BART	24.6	01	15.6
meansum	BART	**34.2**	07.7	20.6
	T5	33.9	**08.4**	**21.2**
	Pegasus	33.8	07.8	20.6

Table 6.10: Domain transfer results evaluated on the FewSum benchmark.

dataset	**model**	**rouge1**	**rouge2**	**rougeL**
combined	BART	**32.8**	**06.5**	**18.6**
	T5	29.2	04.9	16.8
	Pegasus	26.8	03.4	15.6
fewsum	BART	29.6	**06.9**	**18.8**
	T5	27.2	04	15.6
	Pegasus	26.8	03.7	15.2

Table 6.11: Domain transfer results evaluated on the MeanSum benchmark.

PUC-Rio - Certificação Digital Nº 1820996/CA

6.3 Results Discussion

The data shows that training models from scratch demand many resources and take a long time to converge. However, with the realization that recent MRS articles fail to mention large Transformers, we wonder why there was no adoption of the models. Both BART and T5 were released before 2020. With GPT-2 (KDE Group et al., 2018) release, the world became aware of the power of this type of model[1]. They are also readily available in deep learning libraries, Huggingface Transformers [2] an incredible job cataloging and sharing Transformer based models. Their library is easy to use and offers both dataset and custom model repositories, where users can share their experiments with the whole NLP community.

In this environment where powerful language models are widely known by the media and access to these models is easily obtainable using specialized libraries, it is unlikely that the authors of current models did not know about Transformers. This work tries to fill the gap between transformers and MRS by directly comparing current models to transformers. The results indicate that large language models should be on the radar for future research and argue that future models can gain from integrating transformers in developing new approaches.

[1]https://www.theverge.com/2019/11/7/20953040/openai-text-generation-ai-gpt-2-full-model-release-1-5b-parameters

[2]https://huggingface.co

Not only that, the lack of discussion about model size or scaling in papers about MRS only furthers our argument that the field is not connected to or concerned about using large models. As our first experiment showed, it is hard to scale MeanSum architecture. The paper used 4 GPUs to train their model. The only work that mentioned model size was FewSum (Bražinskas et al., 2020). They keep their new model to the same size as the previous model claiming it would be a fairer comparison but don't show how big their model can be scaled and what results the unrestrained model might achieve.

We acknowledge that pre-trained transformers are about ten times larger than the other models explored in this work, and the large parameter count is a double-edged sword. For one side, we have large models that take GPU space and may prove to be a challenge to deploy, and at the same time, it has faster and cheaper training using fine-tuning in a handful of examples.

The deployment of such models is not an issue discussed in any paper about MRS with deep learning models, and the resource requirements to train specialized models are higher than what is need to fine-tune pre-trained Transformers. Both MeanSum and FewSum were trained using multiple GPUs just like the transformers, but transformers were designed to be reused for many downstream tasks. Unlike MRS models, Transformers are also readily available in specialized libraries, making it even easier to pursue a research object using such models instead of relying on each paper's custom code.

Nevertheless, LSTMs are the base model for most previous works on MRS. It is known (Kaplan et al., 2020) that recurrent models do not scale that well when increasing the parameter count or sequence length. Not only that, the practice of scaling up models is not a standard practice methodology of MRS. Even recent models that use Transformers do not provide experiments increasing the model's size or mention scaling properties or limitations their model might have. However, the bleeding edge research in NLP is scaling models to trillion parameters (Fedus et al., 2021) while unsupervised review summarization still works with 50k parameters. It is worth mentioning that current MRS models employ very innovative strategies to make the most of unsupervised data but lacks the number of parameters required to meet scaling laws to achieve optimal performance (Kaplan et al., 2020).

This work's results provide evidence to support that it is time to look at large models and harness their capabilities to capture linguistic information in large unsupervised datasets, suggesting exploring the combination of both strategies to develop better summarization models as a future research path.

PUC-Rio - Certificação Digital Nº 1820996/CA

7 Conclusion

This work explored alternatives to the current research approach on MRS. It extended the base MeanSum architecture with attention, hierarchical attention, and sentiment-based auxiliary tasks, showing that these methods can improve the base results. The experiments also evaluated the use of augmented data to train supervised models on unsupervised tasks, showing that this strategy can deliver similar results to specialized models and harness the power of current supervised summarization models.

This study also showed that fine-tuning large transformer language models achieve better results than current models. Suggesting that the pre-train then fine-tune approach can be used to perform the task using less computational power than training custom models from scratch.

All models were evaluated on ROUGE, the default metric on the field. The experiments also used BERTscore and the Coverage and Density pair as additional metrics. The use of these metrics can bring a better understanding of the results, but none of them is good on their own. The use of many metrics can also lead to an information overload scenario where even with four metrics, it is unclear what model is the best one. The search for a more comprehensive way to evaluate these models is a challenge that is left to future research.

Current developments on MRS shows that the field of MRS is starting to adopt Transformer models as part of modern architectures but is still training every model from scratch. These domain-specific models are excellent at leveraging the characteristics of user reviews, but our results suggest that large Transformers can learn these aspects using the masked language modeling training scheme and large quantities of data.

This work presents evidence to push the adoption of larger models in MRS research to shorten the gap between unsupervised multi-review summarization and general NLP practices by applying pre-trained models on the MRS task. We also discuss expanding the current research framework to accommodate larger transformer models to build unsupervised review summarization systems.

Further exploration of the use of pre-trained models is needed to uncover the full potential of this approach. Likewise, there is also a need to extend the works to other languages other than English. Most research is done using English datasets and the development of other data sources is also left to future work.

PUC-Rio - Certificação Digital Nº 1820996/CA

We conclude this work hopeful about the future of the field. Models are getting better and more accessible every year. Furthermore, even with many challenges, the research community thrives and pushes current architectures' limits, generating summaries with ever-increasing quality.

PUC-Rio - Certificação Digital Nº 1820996/CA

8 References

[Amplayo and Lapata, 2020] Amplayo, R. K. and Lapata, M. (2020). Unsupervised opinion summarization with noising and denoising. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1934–1945, Online. Association for Computational Linguistics. (document), 2.2, 3.1.4, 3.7, 6.1.2

[Angelidis and Lapata, 2018] Angelidis, S. and Lapata, M. (2018). Summarizing opinions: Aspect extraction meets sentiment prediction and they are both weakly supervised. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3675–3686. 2.2

[Ba et al., 2016] Ba, J. L., Kiros, J. R., and Hinton, G. E. (2016). Layer normalization. 3.2.1.1

[Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. cite arxiv:1409.0473Comment: Accepted at ICLR 2015 as oral presentation. 4.1.1

[Bender et al., 2021] Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA. Association for Computing Machinery. 2.2

[Bowman et al., 2016] Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. (2016). Generating sentences from a continuous space. 3.1.2

[Bražinskas et al., 2020] Bražinskas, A., Lapata, M., and Titov, I. (2020). Few-Shot learning for opinion summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4119–4135, Online. Association for Computational Linguistics. (document), 1, 2.2, ??, 3.1, 3.1.5, 5.3, 6.3

[Bražinskas et al., 2020] Bražinskas, A., Lapata, M., and Titov, I. (2020). Unsupervised opinion summarization as copycat-review generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5151–5169, Online. Association for Computational Linguistics. 2.2, 3, ??, 3.1.2

PUC-Rio - Certificação Digital Nº 1820996/CA

[Carlini et al., 2020] Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., Oprea, A., and Raffel, C. (2020). Extracting training data from large language models. 2.2

[Chen et al., 2016] Chen, K.-Y., Liu, S.-H., Chen, B., and Wang, H.-M. (2016). Learning to distill: The essence vector modeling framework. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 358–368, Osaka, Japan. The COLING 2016 Organizing Committee. 2.1

[Chu and Liu, 2019] Chu, E. and Liu, P. (2019). MeanSum: a neural model for unsupervised multi-document abstractive summarization. *International Conference on Machine*. (document), 1, 2.2, 3, ??, 3.4, 4.1.3, 5.3

[Devlin et al., 2019] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). 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)*, pages 4171–4186. 2.2, 3, 3.2, 3.2.2, 4.2, 5.2.2

[El-Kassas et al., 2021] El-Kassas, W. S., Salama, C. R., Rafea, A. A., and Mohamed, H. K. (2021). Automatic text summarization: A comprehensive survey. *Expert Systems with Applications*, 165:113679. 1

[Fabbri et al., 2020] Fabbri, A. R., Kryściński, W., McCann, B., Xiong, C., Socher, R., and Radev, D. (2020). Summeval: Re-evaluating summarization evaluation. *arXiv preprint arXiv:2007.12626*. 5.1

[Fabbri et al., 2019] Fabbri, A. R., Li, I., She, T., Li, S., and Radev, D. R. (2019). Multi-News: a Large-Scale Multi-Document summarization dataset and abstractive hierarchical model. 2.1

[Fedus et al., 2021] Fedus, W., Zoph, B., and Shazeer, N. (2021). Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. 6.3

[Ganesan, 2018] Ganesan, K. (2018). ROUGE 2.0: Updated and improved measures for evaluation of summarization tasks. 1

[Gehrmann et al., 2018] Gehrmann, S., Deng, Y., and Rush, A. (2018). Bottom-Up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4098–4109. aclweb.org. 1

PUC-Rio - Certificação Digital Nº 1820996/CA

PUC-Rio - Certificação Digital Nº 1820996/CA

[Goldstein and Carbonell, 1996] Goldstein, J. and Carbonell, J. (1996). Summarization: (1) using MMR for diversity - based reranking and (2) evaluating summaries. In *Proceedings of a workshop on held at Baltimore, Maryland October 13-15, 1998* -, page 181, Morristown, NJ, USA. Association for Computational Linguistics. 2.1

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`. 4.2

[Grusky et al., 2018] Grusky, M., Naaman, M., and Artzi, Y. (2018). Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719, New Orleans, Louisiana. Association for Computational Linguistics. 1, 2.1, 5.2, 5.2.3

[Hermann et al., 2015] Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701. 1, 3.2

[HochreiterSepp and SchmidhuberJürgen, 1997] HochreiterSepp and SchmidhuberJürgen (1997). Long Short-Term memory. *Neural Comput.* 2.1

[Hong and Nenkova, 2014] Hong, K. and Nenkova, A. (2014). Improving the estimation of word importance for news multi-document summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 712–721. 2.1

[Huang et al., 2014] Huang, X., Wan, X., and Xiao, J. (2014). Comparative news summarization using concept-based optimization. *Knowledge and information systems*, 38(3):691–716. 2.1

[Ioffe and Szegedy, 2015] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France. PMLR. 4.1.3

[Jang et al., 2016] Jang, E., Gu, S., and Poole, B. (2016). Categorical reparameterization with Gumbel-Softmax. 3.1.1

[Joshi et al., 2020] Joshi, M., Chen, D., Liu, Y., Weld, D. S., Zettlemoyer, L., and Levy, O. (2020). SpanBERT: Improving pre-training by representing and

PUC-Rio - Certificação Digital Nº 1820996/CA

predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77. 3.2

[Kaplan et al., 2020] Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020). Scaling laws for neural language models. 6.3

[KDE Group et al., 2018] KDE Group, University of Kassel, DMIR Group, University of Würzburg, L3S Research Center, and (Germany), H. (2018). Language models are unsupervised multitask learners | BibSonomy. `https://www.bibsonomy.org/bibtex/ce8168300081d74707849ed488e2a458`. Accessed: 2020-2-17. 6.3

[Kryscinski et al., 2019] Kryscinski, W., Keskar, N. S., McCann, B., Xiong, C., and Socher, R. (2019). Neural text summarization: A critical evaluation. 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)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics. 1, 2.1, 5.2

[Lebanoff et al., 2018] Lebanoff, L., Song, K., and Liu, F. (2018). Adapting the neural Encoder-Decoder framework from single to Multi-Document summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4131–4141, Stroudsburg, PA, USA. Association for Computational Linguistics. 2.1

[Lewis et al., 2020] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). 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*, pages 7871–7880, Online. Association for Computational Linguistics. (document), 1, 2.2, 3.2.3, 3.10, 4.1

[Liu and Lapata, 2019] Liu, Y. and Lapata, M. (2019). Hierarchical transformers for Multi-Document summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5070–5081, Stroudsburg, PA, USA. Association for Computational Linguistics. 2.1

[Luong et al., 2015] Luong, T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics. 3.2.1.1

PUC-Rio - Certificação Digital Nº 1820996/CA

[Ma et al., 2016] Ma, S., Deng, Z.-H., and Yang, Y. (2016). An unsupervised multi-document summarization framework based on neural document model. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1514–1523. 2.1

[Miao et al., 2020] Miao, Z., Li, Y., Wang, X., and Tan, W.-C. (2020). *Snippext: Semi-Supervised Opinion Mining with Augmented Data*, page 617–628. Association for Computing Machinery, New York, NY, USA. 3.1.3

[Mikolov et al., 2013] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc. 2.1, 2.2

[Miller, 2019] Miller, D. (2019). Leveraging bert for extractive text summarization on lectures. 2.2

[Ni et al., 2019] Ni, J., Li, J., and McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. 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)*, pages 188–197, Hong Kong, China. Association for Computational Linguistics. 2.1

[Nikolov and Hahnloser, 2019] Nikolov, N. I. and Hahnloser, R. (2019). Large-scale hierarchical alignment for data-driven text rewriting. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 844–853, Varna, Bulgaria. INCOMA Ltd. 2.1

[Nikolov and Hahnloser, 2020] Nikolov, N. I. and Hahnloser, R. (2020). Abstractive document summarization without parallel data. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 6638–6644, Marseille, France. European Language Resources Association. 2.1

[Pan et al., 2019] Pan, S., Li, Z., and Dai, J. (2019). An improved textrank keywords extraction algorithm. In *Proceedings of the ACM Turing Celebration Conference - China*, ACM TURC '19, New York, NY, USA. Association for Computing Machinery. ??

[Parida and Motlicek, 2019] Parida, S. and Motlicek, P. (2019). Abstract text summarization: A low resource challenge. 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), pages 5994–5998, Hong Kong, China. Association for Computational Linguistics. 2.1

[Radford et al., 2019] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners. 3.2.3

[Raffel et al., 2020] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67. (document), 1, 2.2, 3.2.4, 3.11, 4.1

[Rogers et al., 2020] Rogers, A., Kovaleva, O., and Rumshisky, A. (2020). A primer in bertology: What we know about how bert works. 2.2, 3.2.2, 4.2, 4.2

[Rossiello et al., 2017] Rossiello, G., Basile, P., and Semeraro, G. (2017). Centroid-based text summarization through compositionality of word embeddings. In *Proceedings of the MultiLing 2017 Workshop on Summarization and Summary Evaluation Across Source Types and Genres*, pages 12–21, Valencia, Spain. Association for Computational Linguistics. 2.2, 3.1.1

[Rush et al., 2015] Rush, A. M., Chopra, S., and Weston, J. (2015). A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Stroudsburg, PA, USA. Association for Computational Linguistics. 1, 2.1

[Schuster and Paliwal, 1997] Schuster, M. and Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.*, 45(11):2673–2681. 1

[See et al., 2017] See, A., Liu, P. J., and Manning, C. D. (2017). 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)*, pages 1073–1083, Stroudsburg, PA, USA. Association for Computational Linguistics. 1, 3.1.2, 4.1, 4.1.1, 4.1.1, 6

[Sennrich et al., 2016] Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Stroudsburg, PA, USA. Association for Computational Linguistics. 3.1.1

PUC-Rio - Certificação Digital Nº 1820996/CA

[Suhara et al., 2020] Suhara, Y., Wang, X., Angelidis, S., and Tan, W.-C. (2020). OpinionDigest: A simple framework for opinion summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5789–5798, Online. Association for Computational Linguistics. (document), 2.2, 3.1.3, 3.5, 3.6

[Szegedy et al., 2016] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826. 5.1

[Tan et al., 2017] Tan, J., Wan, X., and Xiao, J. (2017). Abstractive document summarization with a Graph-Based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1171–1181, Stroudsburg, PA, USA. Association for Computational Linguistics. 2.1

[Varalakshmi K and Kallimani, 2018] Varalakshmi K, P. N. and Kallimani, J. S. (2018). Survey on extractive text summarization methods with Multi-Document datasets. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2113–2119. 1

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. U., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc. (document), 1, 3, 3.2.1, 3.8, 3.9

[Wang et al., 2018] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics. 3.2.2

[Wu et al., 2016] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*. 6.1.3

[Yang et al., 2016] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., and Hovy, E. (2016). Hierarchical attention networks for document classification. In *Pro-*

PUC-Rio - Certificação Digital Nº 1820996/CA

ceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1480–1489, San Diego, California. Association for Computational Linguistics. 4.1.2

[Zhang et al., 2018] Zhang, J., Tan, J., and Wan, X. (2018). Adapting neural Single-Document summarization model for abstractive Multi-Document summarization: A pilot study. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 381–390. 2.1

[Zhang et al., 2020] Zhang, J., Zhao, Y., Saleh, M., and Liu, P. (2020). PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In III, H. D. and Singh, A., editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 11328–11339, Virtual. PMLR. 1, 2.2, 3.2.5, 4.1

[Zhang et al., 2019] Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2019). Bertscore: Evaluating text generation with BERT. *CoRR*, abs/1904.09675. (document), 1, 2.1, 5.2, 5.2.2, 5.2

PUC-Rio - Certificação Digital Nº 1820996/CA