Business Email Abstractive Threads Summarization (BEATS): NLP models reading a chain of 30 emails beats reading them yourself

Matthew Burke
University of Pennsylvania
mburke20@upenn.edu

Aaron Finkelstein
University of Pennsylvania
ajfinky@upenn.edu

Jin-seo Kim
University of Pennsylvania
jins0904@upenn.edu

Sindhura Mente
University of Pennsylvania
smente@seas.upenn.edu

Abstract

Most working professionals spend a significant portion of their time communicating with other members of their organization through email chains, which can easily get overwhelming if these messages are not read soon after they are received. Our project aims to solve this issue by developing an email chain summarizer specifically for the domain of business emails. We started with a T5 model as our weak baseline, prior to fine-tuning it on the Avocado dataset, which contains email threads and example summaries, as our strong baseline, which saw a performance increase over the untuned model. Next, we finetuned a GPT model which appeared to have better results than the baseline models, but was not reflected in our evaluation metrics. To remedy this, we finetuned a T5 model on the Avacado dataset using SummaQA as the evaluation metric, which we present as our final summarizer.

Email Thread:
Subject: lunch this week
Susan: All, Regarding our lunch this week to celebrate the one year anniversaries for Michelle & David, and Mark's birthday, I have a request to make it Wednesday instead of Tuesday. Does anyone have an objection to this? Susan
David: I have another lunch engagement Wed, but I will skip it if everyone else wants to move our lunch. David
Tamra: Susan, Wednesday works out better for me as well. I have a doctor's appointment tomorrow during lunch. Tamra

Short Summary:
Susan emails everyone about an anniversary and offers to change the date. David says he is busy but is willing to go with the majority. Tamra agrees with Susan's date.

Long Summary:
Susan emails everyone about a lunch to celebrate a one year anniversary as well as Mark's birthday. She says she would change the date to a different day. David says he is busy that day with his own appointment but is willing to go with the majority and cancel that appointment to make this one. Tamra agrees with Susan's date as she is busy Tuesday with an appointment.

Figure 1: Email Thread Example from EmailSUM

1 Introduction

Most of us face a barrage of email chains in everyday life. In fact, studies have found that business professionals spend almost one-third of their day on email (Plummer, 2020). Eighty-six percent of professionals prefer email for business communication, and on average, receive at least 200 messages and send around 40 emails per day (Lim, 2020; Acton, 2017). A small improvement in email efficiency, even in terms of minutes or hours per day, can result in enormous corporate-wide financial savings.

A business email thread is not a single document or a dialog; it is an asynchronous branching discussion between multiple writers exchanging information, quoting each other, and succeeding or failing to reach agreements. Email threads are hard to summarize, and a good summary can save a lot of time. The 2021 paper by Shiyue Zhang et al., EmailSUM: Abstractive Email Thread Summarization (Zhang et al., 2021a) is our main reference. We wanted to build a model that can summarize business email threads: who participated, what was discussed, and what were the conclusions or pending questions. Through summarization, the time spent reading emails could be dramatically cut down.

Figure 1 is an example of the general type of problem we are addressing. Our task is to build a generative model that takes in the messages of a business email thread and produces a short summary. The summary should be similar to a human-generated summary and it should contain the key information contained in the thread. This is one of many possible applications of recent progress in NLP to email. We thought this task would be doable and of interest both to us but also as part of a potential product.

2 Literature Review

Zhang et al. (2021a) aims to spur further research on the summarization of conversational threads. To this end, they developed an abstractive Email Thread Summarization dataset, EmailSUM (Zhang et al., 2021b) containing human-annotated short and long summaries of over 2500 email threads of various topics. Their work explores different summarization techniques, employing extractive and abstractive methods, single-document and hierarchical models, and transfer and semisupervised learning methods. They conduct human evaluations of the summarized output on short and long-summary generation tasks, cross-checking results across evaluators to ensure consistency. Finding that commonly used automatic evaluation metrics ROUGE and BERTScore are weakly correlated with human judgments on this email thread summarization task, they urge the NLP community to develop better metrics, as well as emphasize the importance of human evaluation.

Given ROUGE (Research, 2023) and BERTScore (Zhang* et al., 2020) are weakly correlated with human judgments on this email thread summarization task, we look to the findings of DialSummEval: Revisiting Summarization Evaluation for Dialogues (Gao and Wan, 2022), which also explores the limitations to using ROUGE to evaluate the performance of dialogue summarization models. Their paper considers a variety of performance metrics in four dimensions: coherence, consistency, fluency, and relevance, and proposes a unified human evaluation of various models. They then introduce the DialSummEval dataset, containing the output of various models and the corresponding human judgments. They also provide a comprehensive re-evaluation and analysis of the performance of widely used automatic evaluation metrics and models. Gao et al. find that few evaluation metrics are excellent in all dimensions and that BARTScore (Yuan et al., 2021) and QA-based metrics, such as SummaQA (Scialom et al., 2019) and QuestEval (Scialom et al., 2021) are comparatively outstanding and worth exploring.

Automatic text summarization: A comprehensive survey (El-Kassas et al., 2021) provides a comprehensive review of automatic text summarization (ATS) systems. While many surveys focus on extractive summarization, this review offers a broader perspective, emphasizing two lesser-studied ATS approaches. The paper categorizes ATS systems based on input size, summarization approach, output nature, summary language, algorithms used, and summary content, type, or domain. The extractive approach, which scores and selects relevant sentences, is simpler but lacks human-like summarization qualities. In contrast, the abstractive approach uses natural language processing (NLP) for more concise, high-quality summaries but is challenging to develop due to its reliance on extensive natural language generation. The paper also discusses the hybrid approach, blending the two, but notes its limitations in producing quality summaries. The paper concludes by addressing current ATS challenges, urging more research on multi-document, multi-lingual, and multiuser summaries. It points out limitations in input/output formats, supported languages, and a focus on extractive methods. Additionally, it highlights the scarcity of structured training data for deep learning, challenges in maintaining summary quality with compression, and the need for reliable, non-manual evaluation methods.

Text Summarization with Pretrained Encoders (Liu and Lapata, 2019) examines how well pretrained models do at text summarization with and without fine tuning and presents additional methods to improve them for this task. Pretrained language models such as BERT are generally employed in classification tasks, and to use them for text summarization requires a deeper understanding of long-term relationships between words and sentences. The model built by the authors combines a pretrained BERT model with a randomly initialized Transformer decoder using a two-stage approach where the encoder (BERT) is fine-tuned twice, first with an extractive objective and then on the abstractive summarization task. Extractive summarization works by identifying and combining the most important sentences in the given document, and abstractive summarization uses an encoder to map a sequence of tokens in the document to a continuous representation of the document, and a decoder then generates a summary token by token by sampling from this distribution. The authors then tested their final BertSumExtAbs model on three different datasets: CNN/DailyMail containing news articles and highlights, NYT consisting of articles from the New York Times, and XSUM, containing articles paired with a given 1-sentence summary of them. Automatic evaluation was done using

ROGUE scores to determine the informativeness and fluency of the generated summaries. Results showed that the BERT models built upon by the authors were most effective in generating good summaries across all three datasets and outperformed baseline transformer models by a large margin, but note that some of the source text is replicated. They also performed human evaluation using a QA paradigm to compare model-generated summaries against human-generated "gold-standard" summaries and found that human evaluators preferred the BertSumExtAbs model's output against that of other generative models.

3 Experimental Design

The steps involved in the data acquisition and cleaning process are outlined in Figure 2, and follows the data processing steps outlined in the EmailSUM GitHub (Zhang et al., 2021b). Our raw dataset, the Avocado Research Email Collection (Douglas Oard and Golitsynskiy, 2015), was obtained from the Linguistic Data Consortium at the University of Pennsylvania and consists of emails and attachments taken from 279 accounts of a defunct information technology company referred to as 'Avocado'. Most accounts are those of Avocado employees, and the remainder are shared accounts and system accounts.

3.1 Data

Our primary dataset for this project consists of unlabeled email threads with varying lengths. The authors of the EmailSUM paper (Shiyue Zhang et al., 2021) obtained the dataset from the Linguistic Data Consortium, an NLP research group located here at the University of Pennsylvania; by reaching out to professors associated with the consortium on campus, our group was able to bypass the regular $1500 fee and obtain the data, known formally as the "Avocado Research Email Collection" (Douglas Oard and Golitsynskiy, 2015).

To maintain a baseline of comparison with EmailSUM, we preprocess the data in the same manner as the authors of that paper. This process broadly involves anonymizing the dataset, by removing names, email addresses, phone numbers, and more information associated with each email chain. Following, we remove duplicate emails along the thread, as well as reply and forward tags, which solely allows for the message to be extracted and considered by the language models of interest.

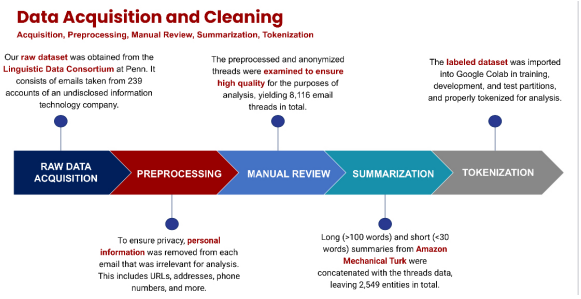

Figure 2: Data Acquisition and Cleaning

Partition	Number of Threads
Training	1,800
Testing	500
Development	249

Table 1: Dataset Partitions

Lastly, we remove chains not written fully in English, that contained less than 3 and more than 10 messages, and that had an overall character count of over 1,000. In total, this leaves 8,116 threads obtained from the original LDC dataset.

The authors then proceeded to utilize Amazon Mechanical Turk to create long (less than 100 words) and short (greater than 30 words) summaries for the threads, resulting in 2,549 data points complete with an associated and verified summary. Within these samples, 1,800 points are used for training, 249 are used for development and 500 are used for testing. In order to minimize the amount of manual labor and costs associated with this project, we decided to use these summaries instead of creating our own. This additionally allows us to directly compare the results of our models with those created by the authors of EmailSUM.

Upon completion of preprocessing, we uploaded the data directly into Google Colab and properly tokenized each partition for analysis.

The full, unfiltered dataset can be viewed in this shared folder, and an example thread can be viewed in Figure 1, located in the "Introduction" section of this report.

3.2 Evaluation Metrics

We selected an evaluation script that takes in two inputs: a system's output and a corresponding set of gold-standard answers. We used this script output to quantify the system's performance in summarization. Following the approach of Shiyue Zhang et al. in their work "EMAILSUM: Abstractive Thread Summarization," we used four ROUGE variants

$$\text{ROUGE-1}_{\text{recall}} = \frac{\text{unigram cand.} \cap \text{unigram ref.}}{|\text{unigram ref.}|}$$

$$\text{ROUGE-1}_{\text{precision}} = \frac{\text{unigram cand.} \cap \text{unigram ref.}}{|\text{unigram cand.}|}$$

$$\text{ROUGE-1}_{\text{F1}} = 2 * \frac{\text{recall} * \text{precision}}{\text{recall} + \text{precision}}$$

$$\text{ROUGE-L}_{\text{F1}} = 2 * \frac{\text{recall} * \text{precision}}{\text{recall} + \text{precision}}$$

Figure 3: ROUGE

$$R_{\text{BERT}} = \frac{1}{|x|}\sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \;, \quad P_{\text{BERT}} = \frac{1}{|\hat{x}|}\sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \;, \quad F_{\text{BERT}} = 2\frac{P_{\text{BERT}} \cdot R_{\text{BERT}}}{P_{\text{BERT}} + R_{\text{BERT}}}$$

$$\hat{R}_{\text{BERT}} = \frac{R_{\text{BERT}} - b}{1 - b}$$

Figure 4: BERTScore

(Lin, 2004), as well as BERTScore (Zhang* et al., 2020) as automatic summarization metrics to evaluate our model.

ROUGE stands for "Recall-Oriented Understudy for Gisting Evaluation" and employs overlap of n-grams between the system and reference summaries, for n = 1 and n = 2 in this study. Additionally, the longest common subsequence is found between summaries. Mathematically, ROUGE is calculated in Figure 3.

BERTScore stands for "Bidirectional Encoder Representations from Transformers Score" where word embeddings of the system and reference summaries are compared with cosine similarity. Each token in the reference is matched to the most similar token in the summary to compute F1-Score. Mathematically, BERTScore is calculated in Figure in Figure 4.

Specifically, we used the following variants for our evaluation criteria:

- **ROUGE-1 (R1)** measures the unigram overlap between the generated and reference summaries.
- **ROUGE-2 (R2)** measures the bi-gram overlap.
- **ROUGE-L (RL)** computes the longest common subsequence (LCS)
- **Summary-level ROUGE-L (RLsum)** computes LCS between each pair of reference and candidate sentences and returns the union-LCS.
- **BERTScore** measures semantic similarity by aligning the candidate and reference summaries and then computing a unigram F-1 score.

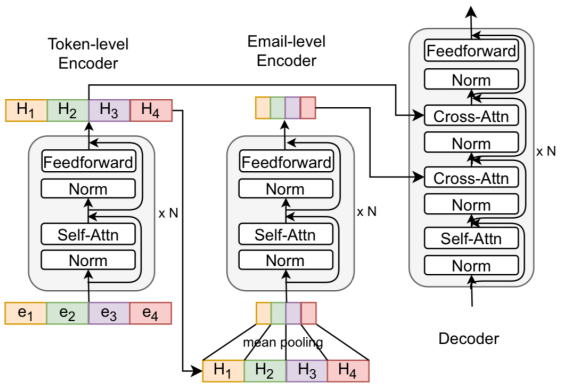

Figure 5: The architecture of our hierarchical T5.

These metrics were averaged across long and short summaries to provide a general metric on how each model performs across different specifications.

3.3 Simple Baseline

Shiyue Zhang et al. present over a dozen baseline models, from which we pull two for our weak and strong baselines: T5 unfinetuned and T5 finetuned. We selected T5 (Raffel et al., 2023), which is a Transformer (Vaswani et al., 2023) based seq-to-seq model pretrained with large-scale English data. According to Zhang, T5 performs well on many NLP tasks, and Zhang et al. provide the script, making it straightforward to replicate.

4 Experimental Results

In summary, T5 unfinetuned unequivocally produced summaries with the worst evaluation, while T5 finetuned and GPT-3.5 Turbo finetuned had roughly equivalent performance, in regard to the ROUGE and BERTScore metrics of interest. GPT 3.5 slightly outperformed T5 finetuned in BERTScore Recall and F1 and ROUGE-1 and L, while the latter had strongest performance in ROGUE-2. Upon review of the generated summaries, it becomes clear that neither of these metrics entirely capture the ability of a summary to describe the full semantic meaning expressed in a given thread.

	Weak Baseline T5	Strong Baseline T5 fine-tuned
Rouge1	0.1886	0.3041
Rouge2	0.04235	0.09586
RougeL	0.1416	0.2391
RougeLsum	0.1440	0.2389
BertSc Prec	0.8431	0.8855
BertSc Rec	0.8434	0.8715
BertSc F1	0.8428	0.8783

Table 2: Evaluation Metric Results

As our main baseline, we take the email thread as a single document and finetune a T5 base to generate the summary (T5base). A similar setup is also used in transfer and semi-supervised learning:

- **T5 unfinetuned (i.e., Weak Baseline)**: In the baseline performance, we find that our weak baseline, unfinetuned T5, is more extractive and pulls strings of text verbatim from the original email chain:

- **T5 finetuned (i.e., Strong Baseline)**: In the baseline performance, we find that our strong baseline, finetuned T5, is more abstractive and synthesizes new sentences summarizing the email chain

Since our training dataset is small, we find that using the pretrained knowledge transfer is crucial. Training a T5 model from scratch performs poorly.

Our weak baseline was evaluated with these metrics Table 2. By every metric, the strong baseline outperformed the weak baseline. Further, manual inspection of a few examples indicated that the strong baseline was creating a more coherent and plausible summary than the weak baseline (whether or not it matched the email thread).

Given that the same strong baseline was utilized by the EmailSum authors (Zhang et al., 2020), we achieved nearly identical performance in each evaluation metric of interest.

The following is an example of the model output given a specific email thread:

- **Thread:** Subject: 1:30pm conference call—Ty: Dan, Can you sit in on a call with Accenture today at 1:30pm to talk about Symbol devices, barcoding into an application, and a few other questions regarding an opportunity at Visa? Thanks, Ty

  Dan: Yes but need more tech info on our symbol peripheral capabilities so may need Amitabh.

  Ty: I think you may be able to cover this initial discussion without Amitabh. We'ss dial you in or come over to Amit's office. Ty

  Dan: I'm working from home today, so you can call my mobile of give me a dial-in ph#. Dan USERNAME@DOMAIN.COM Director of Sales Engineering PHONENUMBER m HTTP://LINK w

  Dan: are you going to call my mobile phone or do you have a dial-in ph# ? Dan USERNAME@DOMAIN.COM Director of Sales Engineering PHONENUMBER m HTTP://LINK w

- **Human summary:** Ty asks Dan if he can take a conference call with Accenture today at 1:30pm. Dan acknowledges he can and is working from home.

- **Weak baseline**: Dan USERNAME@ DOMAIN.COM: i'm working from home today, so you can call my mobile or dial-in ph# .

- **Strong baseline**: Ty asks Dan if he can sit in on a conference call with Accenture today at 1:30pm to talk about Symbol devices, barcoding into an application, and a few other questions regarding an opportunity at Visa.

4.1 Extensions

Extension 1 was based on GPT3.5 and Extension 2 was based on SummaQA, as outlined below:

4.1.1 Extension 1

In order to enable a direct comparison between the model performances of our strong baseline and GPT3.5 extension, we employed the same evaluation measures as our second milestone, which includes the four variants of ROUGE metrics (Lin, 2004), and mean BERTScore across the short test dataset (Zhang et al., 2020). We fine-tuned a GPT 3.5 turbo model on the identical training and validation sets used for tuning the T5 strong baseline. The following prompt, together with the original email thread and a corresponding human summary, was included in each training example: "You are an assistant who generates a short summary of a thread of email chains. Please be informative, while containing all the important details about

	GPT 3.5	Strong Baseline T5 fine-tuned
Rouge1	0.3376	0.3041
Rouge2	0.07542	0.09586
RougeL	0.2406	0.2391
RougeLsum	0.2418	0.2389
BertScore Precision	0.8804	0.8855
BertScore Recall	0.8813	0.8715
BertScore F1	0.8808	0.8783

Table 3: Evaluation Metric Results

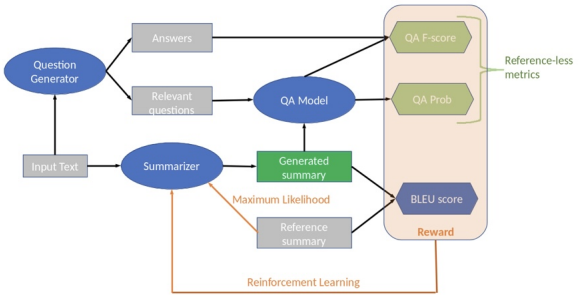

Figure 6: SummaQA Evaluation Methodology

Metric	Weak T5 Baseline	Strong T5 Baseline	GPT3.5	SummaQA-finetuned T5
Rouge1	0.1886	0.3041	**0.3376**	0.3336
Rouge2	0.04235	0.09586	0.07542	**0.1060**
RougeL	0.1416	0.2391	0.2406	**0.2755**
RougeLSum	0.1440	0.2389	0.2418	**0.2752**
BertScore Precision	0.8431	0.8855	0.8804	**0.8856**
BertScore Recall	0.8434	0.8715	0.8813	**0.8871**
BertScore F1	0.8428	0.8783	0.8808	**0.8862**

Figure 7: Model Comparison

who are the main speakers and what is the topic being discussed." By every metric, the GPT3.5 performance was on par with, or slightly outperformed, the strong T5 baseline. The performance differences were most pronounced in Rouge1, Rouge2, and BERT Recall scores, whereas differences in other metrics were more subtle, as shown in Table 3.

Please refer to section 4.1.3 for an illustrative comparative example of model output.

4.1.2 Extension 2

For extension 2, we applied SummaQA, which is visualized in Figure 6. SummaQA (Scialom et al., 2019) applies a BERT-based question-answering model to answer cloze-style questions using generated summaries. Questions are generated by masking named entities in source documents associated with evaluated summaries. The metric reports both the F1 overlap score and QA-model confidence. We can either compare the summary to the original thread (unsupervised) or we can compare the summary against the human-generated summaries (supervised). We applied SummaQA in two ways.

First, we evaluated the baseline model output and the output from Extension 1, verifying that although the ROUGE scores were similar, the output from GPT contained more of the key pieces of information than the summaries from T5 fine-tuned to improve the ROUGE score. Second, we fine-tuned T5 using the SummaQA score as an objective. This increased the SummaQA score further, although it is unclear whether this increase represented a real improvement in quality.

In Figure 7 we see that SummaQA fine-tuned T5 outperforms on almost all metrics, but those metrics do not provide the full story, as explained in the following illustrative comparative example of model output, in the following section.

4.1.3 Model Output

The following is an example of the model output (move to scoring.md): Original email thread:

Subject: lunch this week————Susan: All, Regarding our lunch this week to celebrate the one year anniversaries for Michelle David, and Mark's birthday, I have a request to make it Wednesday instead of Tuesday. Does anyone have an objection to this? Susan————David: I have another lunch engagement Wed, but I will skip it if everyone else wants to move our lunch. David————Tamra: Susan, Wednesday works out better for me as well. I have a doctor's appointment tomorrow during lunch. Tamra

- **Human Summary:** Susan emails everyone about an anniversary and offers to change the date. David says he is busy but is willing to go with the majority. Tamra agrees with Susan's date.
- **T5 Weak Baseline:** lunch this week to celebrate the one year anniversaries for Michelle & David, and Mark's birthday . does anyone have an objection to this?
- **T5 Strong Baseline:** Susan has a request to make lunch this week to celebrate the one year anniversaries for Michelle David, and Mark's birthday. David has another lunch engagement wed, but I will skip it if everyone else wants

- **Extension 1: Fine-tuned GPT:** Susan asks coworkers if they can move a lunch to Wednesday. David and Tamra say they can and list their reasons why.

- **Extension 2: T5 fine-tuned to SummaQA:** Susan asks for a Wednesday lunch to celebrate the one year anniversary of Michelle & David and Mark's birthday. David says he will skip it. Tamra

In looking at the outputs, the Weak Baseline is extractive, with verbatim copies of two phrases from the input. The Strong Baseline is abstractive, but did not consistently switch to third person, and left out the conclusion. Extension 1 is close to the human summary, though more focus on what than why. Extension 2 provides an efficient abstractive summary of first email, keeping extra information, but provides an incorrect summary of responses.

4.2 Error Analysis

One type of error commonly made by the weak baseline, but also made by the other models less frequently, was to truncate the email thread rather than to summarize it. A verbatim copy of the first few sentences of the first email message might show the subject of the thread and might score well by some metrics, but it would not show the back-and-forth discussion and the conclusion.

Two other types of errors were (1) Failing to understand the sender's intent and (2) Failing to identify the roles of the sender and receiver. Figure 8 provides a good illustration of these concepts (source: Table 6 in EmailSUM):

- In terms of failing to understand the sender's intent, in the first example, Om intends to summarize the important points from a meeting, while the model only picks the first piece of detail in that email as the summary (this error occurs as extraction is biased to the first sentence of each email).

- In terms of failing to identify the roles of the sender and receiver, in the second example, the model wrongly takes "2 fixes in 382 are in the patch installer" as information provided by Nilesh, whereas it is supposed to be by Diana.

Another type of error is to describe the actions ("a meeting was scheduled") but leave out key pieces of information (When? Where?). The

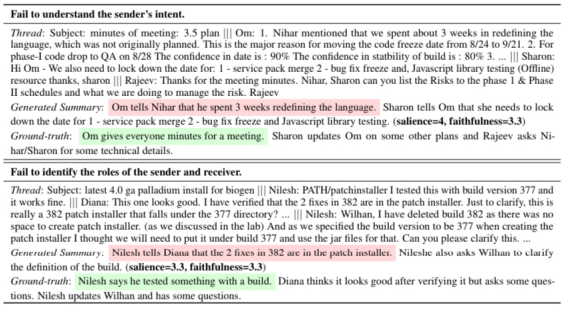

Figure 8: Example of Types of Errors from EmailSUM

SummaQA metric is sensitive to this type of error, which was made more often by GPT than T5 fine-tuned to SummaQA.

An important error consideration relates to all of the models: the training data itself. Given the cost, labor, and time intensive process of sourcing the emails and providing human-made summaries, only 2,459 observations were available for analysis in the end, with only 1,800 used for training and testing. It is quite possible that this number is insufficient for optimal performance on this task for both fine-tuned models. Upon searching online, there are not many published articles on HuggingFace (a) defining an exact number needed to finetune an LLM for summarization tasks such as this one, although users on HuggingFace (b) forums seem to use over 20,000 when available. Another extension would be to perform the same evaluation given a much larger training corpus, and publish results defining an optimal number of examples before performance plateaus.

5 Conclusions

Throughout this project, we developed a couple of different models for the task of summarizing business email threads.

First, we used a base T5 model as a baseline for performance, and it performed badly as expected. Then we finetuned the T5 model on the Avacado dataset and saw a performance increase as presented in the results above, though the generated summaries seemed to be extractive and copied phrases from the text verbatim rather than produce a general summary of what was said. This achieved performance similar to the models presented in the EmailSum paper.

After this, we finetuned a GPT model which appeared to be more abstractive and produced better summaries from a human perspective, however this

was not consistent with our BERT and ROGUE evaluation metrics, which seemed to suggest that the performance of the GPT and finetuned T5 models were comparative. This made it apparent that we needed a stronger evaluation metric, so we decided to use SummaQA.

First, we evaluated our strong baseline and our GPT model using this metric and found that GPT performed better. Next, we finetuned a T5 model using SummaQA as the objective and found that the performance of this model was better than the prior three, and also better than the models developed by the authors of the EmailSum paper, suggesting that state-of-the-art performance was achieved; however, it was unclear to us whether this was due to the evaluation metric itself, or if the model finetuned with SummaQA was able to better learn how to summarize email chains. Overall, we found this to be a valuable learning experience

6 Acknowledgements

We would first like to thank Prof. Mark Yatskar and the TA's for helping us throughout the semester, and especially Yufei Wang, our project mentor, for meeting with us frequently and guiding us through the project as well as giving us valuable feedback and for being a sounding board for our ideas. We are incredibly grateful for all of you!

We would also like to extend our thanks to Shiyue Zhang et al. for inspiring us to pursue this project, and for pointing us to the Avocado Dataset with a rich set of email threads and labled summarizations. Additionally, the work done by Mingqi Gao et al, El-Kassas et al, Lapata, and Liu for providing more detail about text summarization and the limitations of current evaluation methods in their papers. Finally, we would like to thank Scialom et al for their work in developing SummaQA, which we took a lot of inspiration from for our second extension.

We are also grateful to Prof. Mark Liberman and the Linguistic Data Consortium at the University of Pennsylvania for providing us with access to the Avocado Dataset free of charge, as without this, we would have been unable to pursue this project.

References

Annabel Acton. 2017. How to stop wasting 2.5 hours on email every day.

David A. Kirsch Douglas Oard, William Webber and Sergey Golitsynskiy. 2015. Avocado research email collection.

Wafaa S. El-Kassas, Cherif R. Salama, Ahmed A. Rafea, and Hoda K. Mohamed. 2021. Automatic text summarization: A comprehensive survey. *Expert Systems with Applications*, 165:113679.

Mingqi Gao and Xiaojun Wan. 2022. DialSummEval: Revisiting summarization evaluation for dialogues. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5693–5709, Seattle, United States. Association for Computational Linguistics.

HuggingFace. a. Summarization.

HuggingFace. b. Thoughts on quantity of training data for fine tuning.

Ariel Lim. 2020. 2020 email marketing statistics to convince you to grow your email list.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders.

Matt Plummer. 2020. How to spend way less time on email every day.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

Google Research. 2023. Python rouge implementation.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, Jacopo Staiano, Alex Wang, and Patrick Gallinari. 2021. QuestEval: Summarization asks for fact-based evaluation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6594–6604, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thomas Scialom, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2019. Answers unite! unsupervised metrics for reinforced summarization models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation.

Shiyue Zhang, Asli Celikyilmaz, Jianfeng Gao, and Mohit Bansal. 2021a. Emailsum: Abstractive email thread summarization.

Shiyue Zhang, Asli Celikyilmaz, Jianfeng Gao, and Mohit Bansal. 2021b. Emailsum: Abstractive email thread summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert.