GitHub Link

Problem Statement

What is communication overload?

In addition to an increasingly-digital age, work-from-home options have become very popular post-COVID—individuals who may work together closely in org. structure can be physically on two sides of the globe. Nevertheless, communication needs have not really changed—if anything, expected response-times have dramatically gone down. As a result, programs/platforms like Discord, Slack and (Microsoft) Teams have become must-have's for almost every business, replacing, at least for internal communications, traditional email.

Whether these are features or bugs—depending on whom you might ask—this presents some new challenges and accentuates some old ones. Such as:

- **Messiness**: Unstructured, Decentralized, Unfiltered. For members trying to find communications on a topic, these chat logs can be very daunting to wade through. Search options can obviously help but the nature of these chats make it difficult to capture the entire conversation since replies may not come in a sequential manner and may not be tied to the original thread. Individuals discussing a topic might have their chat "interrupted" in the log (unlike emails which, generally, have replies directly under the original, no matter how long it took to respond). Most platforms offer organizations the ability to establish "channels" which can be dedicated to topics or organizations' teams. However, there is a significant tradeoff since specialization requires members to monitor more channels.
- **Miscommunication**: Less formal communications can create confusion between employees; quick replies may not be as well-thought through as longer-form emails; they may also be duplicative; it also easy to miss follow-up communications that might offer updates/corrections without significantly diligent monitoring.
- **Productivity**: Having to spend time wading through hundred and hundred lines of a chatroom and/or the need for constantly monitor chats and the fear of missing key communication will likely force employees to be less efficient at performing their core responsibilities.
- **Stress**: Altogether, it is not surprising that information overload contributes to more rote work for employees to try to handle all their relevant communications and to a greater sense of stress about the possibility of overlooking a key piece of information.

Gartner has done annual research based on surveys which can help quantify these issues and their impacts.[1] For instance, Harvard Business Review reported that Gartner's 2022 survey (of ~1k employees/managers) found that

[1] This data appears to be behind a paywall but some tidbits can be found in articles on this topic.

- 38% described the volume of communications in their organization as “excessive”;
- 27% described this as “at least” somewhat overwhelming
- ONLY 6% of those who reported being overwhelmed also said that they were “highly likely” to stay with the same organization.

Based on these numbers, a Harvard Business Review article discussing the survey concluded that employees wasted 3.5 hrs each week dealing with information burden. A 2025 article from Gartner reported that in most recent survey on the topic, 43% of employee’s day is spent on consuming information which they translated to ~$4.6mm spent per week (assuming an organization of 10k members). The article also states that 25%+ of employees and 38% of managers reported being overwhelmed by the excessive communications.

There are some straight-forward approaches to curtail these problems. For instance, training employees on how to communicate with one another on these platforms—encouraging formal conversation and coming up with keywords to help indicate type/topic of message you are sending—but a) such practices may defeat the purpose of using such a platform, b) do not solve the problem of lacking a well-structured database of communications, and c) does not entirely address the productivity/stress concerns.

It is therefore reasonable to pursue the possibility of using NLP/AI tools in order to keep the obvious benefits of using these communication platforms while, at least, curtailing the negative impacts.

Project Approach

SAMSum (Samsung Abstractie Messenger Summarization)

This database contains 16k+ conversations which were created by a group of linguists to reflect online conversations they would normally have and were meant to cover any number of topics that they might discuss over the course of a regular day—hopefully, per the paper’s authors, this will avoid domain specificity. These conversations were then passed on to another group of language experts who then annotated them under the following constraints:

- (relatively) Short
- Extract important pieces of information
- Include the speakers’ names
- Written in 3rd person

The conversations can be further organized based on the number of utterances:

1. 3-6
2. 7-12
3. 13-18
4. 19-30

And by the number of speakers—typically (~75%) two but occasionally 3+.

What about chat summation makes the problem unique?

Assuming a "*[Speaker_1]: [Message_A] /n [Speaker_2] : [Message_B] /n*" syntax[2], the chat format offers some helpful structure as it is easy to understand at least two persons of interest that the algorithm can use to generate a summary. But beyond that, there is a fair bit of nuance that can be captured. Let's look at an example from the SAMSum database:

Amanda: I baked cookies. Do you want some? **Jerry**: Sure! **Amanda**: I'll bring you tomorrow :-)	**Amanda** baked cookies and will bring **Jerry** some tomorrow.

First, we can see some of the difficulties when dealing with chat logs:

- Informal/Unique Language: We see the smiley face used as well as the slightly confusing grammar by Amanda.
- Pronoun usage: A successful model will need to recognize and handle pronouns like 'I' and 'you' which should be assigned to one of the speakers.
- Dispersed Info: The provided summary includes information dispersed through the conversation, including the name of the other speaker, whose utterance(s) otherwise does not contribute to the summary.
- Context vs. Concise: The human-designed summary focuses on what has and will be happening; however, I think you could argue it also misses key information—specifically, Jerry's affirmation. Depending on the business's needs, one might prefer and different summation.

Why BERT?

We will want to leverage the BERT-based encoder architecture in order to generate representations of the dialogues and their given summaries that

A BERT-based encoder-decoder architecture is a neural network framework commonly used for sequence-to-sequence tasks in Natural Language Processing (NLP). It leverages the power of the BERT model, primarily as an encoder, to process input text and then uses a separate decoder to generate an output sequence.

The BERT encoder's role is to create rich, contextualized representations (embeddings) of the input text. BERT is pre-trained on large amounts of text data using tasks like masked language modeling and next sentence prediction, allowing it to capture intricate relationships and meanings within sentences and across sentence pairs. When used as an encoder, it takes the input sequence and produces a sequence of hidden states, where each state represents the corresponding input token's context-aware embedding.

[2] Per the introductory research paper, this is the case: "we [the authors] specified a format…a colon should separate an author of utterance from its content, each utterance is expected to be in a sperate line.

The decoder's function is to take these contextualized embeddings from the encoder and generate the output sequence, token by token. Decoders are typically auto-regressive, meaning they generate each output token based on the previously generated tokens and the encoder's output. This allows the decoder to build the output sequence sequentially, maintaining coherence and relevance to the input.

This architecture is particularly well-suited for sequence-to-sequence problems in NLP, such as machine translation, text summarization, and question answering. The encoder effectively understands the input sequence, and the decoder translates this understanding into the desired output sequence, making it a powerful combination for complex language tasks.

Why auto-regressive w/ GPT?

Auto-regressive modeling is a statistical concept where a model predicts the next value in a sequence based on the values that precede it. In simpler terms, each prediction is conditioned on the previous predictions or observed values in the sequence.

In the context of Natural Language Processing (NLP) and sequence generation tasks like text generation, machine translation, or dialogue summarization, auto-regressive modeling plays a crucial role. The model generates the output sequence one token (word or sub-word) at a time. To predict the next token, the model considers the input sequence (if applicable, like in translation or summarization) and all the tokens it has generated so far in the output sequence. This sequential dependency ensures that the generated text is coherent and contextually relevant.

Common mechanisms used in auto-regressive models for sequence generation include using the previously generated output token as an input to the model for predicting the next token. This feedback loop allows the model to build the output sequence incrementally. Often, attention mechanisms are also incorporated to help the model focus on relevant parts of the input sequence and the previously generated output when making the next prediction.

Fine-Tuning Advantages

BERT models are very impressive tools but can be very resource-intensive, especially in the initial training stage where BERT models require an extensive corpus to be truly optimized. Thanks, in large part, to HuggingFace, we have access to a number of pre-trained BERT models that can be considered

Optimization & Potential Issues?

1. Dialogues in SAMSun are multi-turn conversations with multiple speakers. Capturing the nuances of speaker turns, coreference resolution across utterances, and the informal nature of spoken language can be challenging. BERT's ability to handle long sequences is relevant, but explicitly modeling speaker information or turn changes might be necessary for optimal performance. The auto-regressive decoder needs to maintain coherence and track who said what to produce an accurate summary.

2. SAMSun provides abstractive summaries, meaning the summaries often contain phrases or sentences not directly present in the original dialogue. Training an auto-regressive model to generate truly abstractive summaries, rather than simply extracting and rearranging sentences, can be difficult. Ensuring the model generalizes well and doesn't overfit to the training data's abstractive style is a key consideration.

Measures of Success

Why ROUGE?
ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a class[3] of metrics that are popular to test for quality of text summarization (among other things). Generally speaking, they test for gram overlaps between the generated summer and the provided one with higher scores indicating more overlap and therefore a better model. As the long-form name suggests, it focuses on recall scores—i.e. how many words in the generated-summary are also in the reference/target summary? This means we ignore precision—the ratio of words in the generative summary that are in the target vs not in the target.

However, according to the paper introducing the SAMSum corpus, the authors found that ROUGE scores were overly optimistic when measuring the dialogue summary (vs. traditional news articles). The concern is that there are sematic nuances in dialogues that do not lend itself to being measured by simple overlap measures. The paper concludes that further research is needed to find a better ROUGE score so that the training process can be improved.

How can human judgement can supplement?
The original paper was able to leverage their resources to have linguist rate the generated summaries based on the dialogue alone (i.e. did not consider the human-provided summary). This is a great use of resources but unfortunately, I do not believe that time will allow for such a deep dive.

Business-Specific Metrics to Consider?
Various articles and blogs offer similar information about how to diagnose a problem of information overload and to assess how well treatments work. This includes:

- **Employee engagement:** Monitor participation in meetings and response rates to critical updates.
- **Productivity levels:** Assess whether employees can complete tasks more efficiently.
- **Feedback surveys:** Gather qualitative data on employee satisfaction with communication processes.

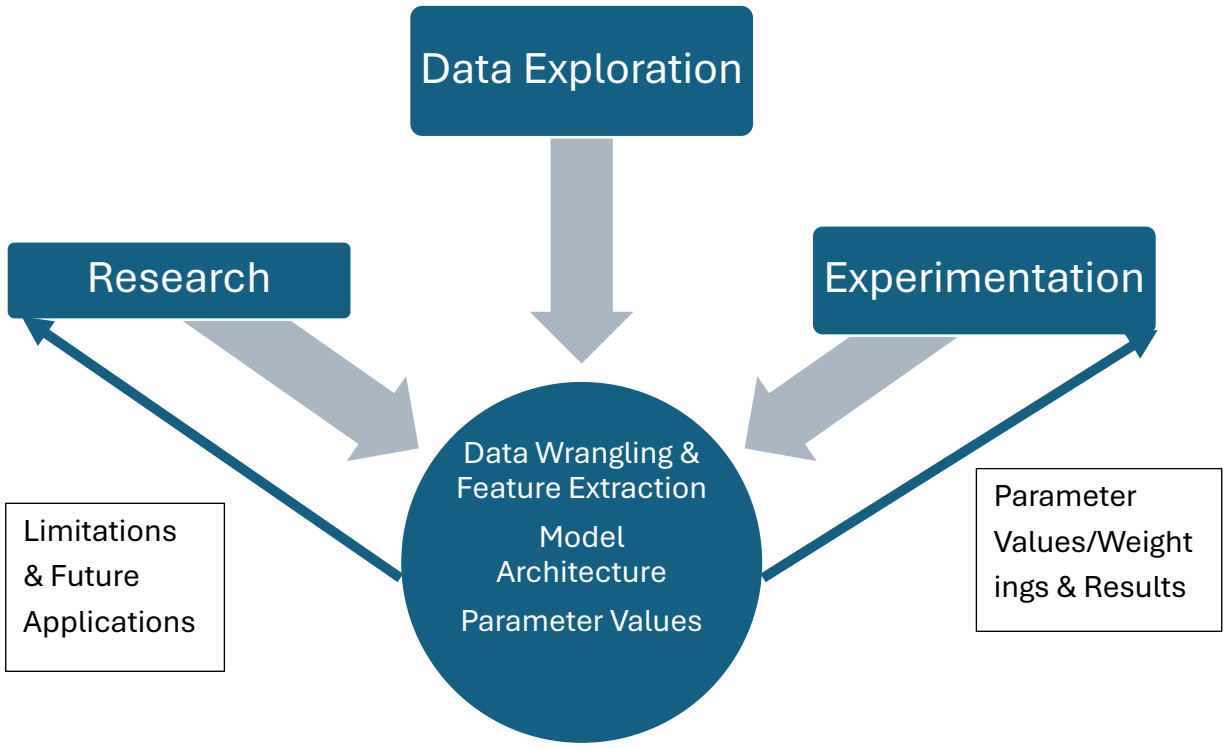
In addition, I believe it would be interesting to consider introducing some more complex test examples to see how the model choice handles unique issues that the company is aware of based on human experience. This could include:

[3] It appears that there are five metrics in this family that focus on the size of the n-grams and how they are chosen/weighted.

- Unique Vocabulary
- Length of Conversations
- # of Topics Covered

Process & Timeline

1. Prepare the input data by tokenizing each conversation
2. Choose a pre-trained BERT model and experiment with other parameter values in order to take the tokenized dialogue and generate embeddings (contextualized vector representations).
3. Build an autoregressive model that will take the BERT's output sequentially and build a summarization based on the dialogue's representation and the output it has already generated at any given step of the process.
4. Compare the generated summary to the given-human summary for scoring
5. Continue steps to maximize a score (or minimize some error)
6. Once complete, consider additional training based on the model's ability to address company-specific needs.



Choices

- Database Split
- BERT Model (Pre-Trained)
- BERT Parameters
- GPT Options
- Base Metric
- Custom Business-Specific Measure of Success
- Additional Dataset / Extension for Domain-Specific Needs;

Timeline

- Thursday/Friday (7/30 & 8/1): Build Repo + GitHub; Begin Proposal; Organize Research Works Thus Far;

- Saturday/Sunday (8/2 & 8/3): Submit Proposal; Finish Research on Dataset, Communication Overload, pre-Trained Model Options & Begin Data Exploration of SAMSum;
- Sunday (8/3): Build Notebook w/ Outline of Basic Steps;
- Monday/Tuesday (8/4 & 8/5): Programming Data Exploration & Decisions on Modeling Approach; Choose Ultimate Dataset (including the possibility of custom conversations as part of validation/test set(s));
- Wednesday (8/6): Implement Model in Code & Consider Results/Debugging Needs; Finalize Model; Collect Output for Analysis
- Thursday (8/7): & Consider Options for Summarizing; Secondary Deliverables; Consider Future Extensions
- Friday (8/6): Other Capstone to-dos.
- Saturday (8/7): Finalize Everything for Submission; Check for Consistency Across Deliverables.

Research Papers

On Communication Overload & Automative Summation

- https://tchop.io/resources/glossary/internal-communication/information-overload
- https://pmc.ncbi.nlm.nih.gov/articles/PMC10322198/#:~:text=More%20specifically%2C%20Klapp%20(1986),processing%20capacity%20of%20the%20recipient.
- Harvard Business Blog on Reducing Overload
- https://www.ebsco.com/research-starters/library-and-information-science/information-overload#:~:text=Information%20overload%20refers%20to%20the,enhance%20focus%20and%20reduce%20stress.
- Gartner 2025 Reduction Strategies

SAMSum Database

https://huggingface.co/datasets/knkarthick/samsum

ROUGE Score

ROUGE Wiki

https://medium.com/@eren9677/text-summarization-387836c9e178

https://aclanthology.org/W04-1013.pdf

ROUGE Matrix Blog

HuggingFace Review of ROUGE

ROUGE for Python

BERT Models

BERT Wiki

Shortening Strategies in BERT Models

Classifying Online Conversations with QiBERT

BART Models

Schneppat Blog Overview.

Overview II w/ HF Example

Autoregressive vs Autoencoding Blog