Large-scale Summarization of Chat Transcripts in the Absence of Annotated Summaries

Pratik K. Biswas
Artificial Intelligence and Data, Verizon Communications
Basking Ridge, New Jersey, USA 07920
pratik.biswas@verizonwireless.com

Abstract

Text summarization is the process of condensing a piece of text to fewer sentences, while still preserving its content. Chat transcript, in this context, is a textual copy of a digital/web conversation between a customer (caller) and agent(s). This paper presents a locally developed hybrid method that combines extractive (unsupervised) and abstractive (supervised) summarization techniques in compressing ill- or un-punctuated chat transcripts to produce more readable summaries. Extensive testing, evaluation and comparisons have demonstrated the efficacy of this approach, in the absence of annotated (reference) summaries, for large-scale summarization.

1 Introduction

Automatic document summarization aims to compress a textual document to a shorter, more informative format while keeping key information of the original text. Numerous approaches have been developed for automatic text summarization and can be broadly classified into two groups: extractive and abstractive summarization. Extractive summarization extracts important sentences from the original text and reproduces them verbatim in the summary, while abstractive summarization generates new sentences. Hybrid Summarization attempts to combine these two approaches in some form.

Chat transcription is defined as the process of converting a digital or web conversation into written words to be stored as plain text in a conversational language. In this paper, however, we will be confining ourselves to *textual descriptions of web chats between customer (caller) and agent(s) (customer representatives) of a* ***phone company***. Automatic summarization of chat transcripts, in this context, pose certain unique challenges, as follows: 1) they are not continuous texts but include conversations between customers and agents, 2) they are often very short or very long, and can include a large number of sentences that are irrelevant and even meaningless, 3) they include several ill-formed, grammatically incorrect sentences, 4) they are either un- or improperly punctuated, 5) there is a dearth of a large collection of human-crafted annotated (reference) summaries that can be used as training samples and 6) existing open-source summarization tools don't perform well with chat transcripts unless properly customized or fine-tuned.

In this paper, we have presented a hybrid summarization technique that combines extractive summarization, comprising of *channel separation* (separation into customer and agent transcripts), *topic modeling, sentence selection* and *punctuation restoration* with *supervised* abstractive summarization via *transfer learning* to produce properly punctuated, fixed-length and readable customer and agent summaries, from the original chat transcripts, that can adequately summarize customer concerns and agent resolutions.

2 Related Work

Related research can be broadly grouped into three categories: 1) extractive, 2) abstractive and 3) hybrid Summarization.

Radev et al. (2002) defined summary as "a text that is produced from one or more texts, that conveys important information in the original text(s), and that is no longer than half of the original text(s) and usually, significantly less than that." Automatic text summarization gained attraction as early as the 1950s. Different methods and extensive surveys of automatic text summarization have been provided in (Zechner,

1997; Mani, 2001; Jones, 2007; Jezek and Steinberger, 2008; Nenkova and McKeown, 2012; Saggion and Poibeau, 2013).

Luhn (1958) introduced a method to extract salient sentences from the text using features such as word and phrase frequency. Gong and Liu (2001) and Wang et al. (2009) summarized multiple documents using topic models. Miller (2019) used Bidirectional Encoder Representations from Transformers (BERT) for summarization of lecture notes. Liu (2019) described BERTSUM, a simple variant of BERT, for extractive summarization. Liu and Lapata (2019) showcased how BERT could be generally applied in text summarization and proposed a general framework for extractive and abstractive models. Feigenblat et al. (2021) introduced TWEETSUM, a large-scale database of customer support dialogs with extractive and abstractive summaries along with an unsupervised extractive summarization method, specific to these dialogs.

Lin and Ng (2009) and Khan and Salim (2014) reviewed the various methods for abstractive summarization. Nallapati et al. (2016), Paulus et al. (2017), See et al. (2017) and Liu et al. (2017) employed recurrent neural networks, deep reinforcement learning, pointer-generator and generative adversarial networks for abstractive summarization. Lewis et al. (2019) introduced BART, a denoising autoencoder for pre-training sequence-to-sequence models that was particularly effective when fine-tuned for text generation (e.g., abstractive summarization, translation, etc.). Beltazi et al. (2020) presented Longformer, useful for long document summarization. Tuggener et al. (2021) provided an extensive overview of existing dialog summarization data sets and mappings from data sets to linguistic models. Fabbri et al. (2021) crowdsourced four new datasets from news comments, discussion forums, community question answering forums, as well as email threads and benchmarked state-of-the-art abstractive summarization models on their datasets. Zhong et al. (2021) presented DialogLM, a pre-trained neural encoder-decoder model for long dialog understanding and abstractive summarization.

Bae et al. (2019) followed a hybrid architecture, rewrote sentences from a document and then paraphrased the selected ones to generate a summary. Su et al. (2020) combined the two summarization methods to generate a variable-length, fluent summary.

3 Major Contributions

Our main contributions and advantages can be summarized as follows:

1. We have integrated topic modeling and embedding based sentence selection with transformer (BERT) based punctuation restoration for extractive summarization through a 10-step sequential procedure.

2. We restore punctuation in the summaries of un-punctuated or ill-punctuated transcripts.

3. We have fine-tuned powerful, transformer-based language models, on locally extracted summaries, for abstractive summarization of chat transcripts through transfer learning.

The summaries can be useful both as historical records and reminder messages of prior chats.

4 Hybrid Summarization of Chat Transcripts

We propose a hybrid strategy that includes extraction, fine-tuning, and abstraction. Its main objective is to provide a hybrid summarization framework that can first extract the summaries of transcripts to create a large enough training sample, and then use this sample to fine-tune pre-trained language model based abstractive summarizers to generate new summaries of unseen transcripts through transfer learning. The resultant summaries are expected to be at least as good as the extractive summaries, with the tacit expectation that the pre-training encoded in the abstractive approach would make the summaries even more fluent, coherent and help reduce some grammatical errors found in the original transcripts. So, the strategy involves 2 sequential phases. Phase I uses an extractive summarizer, while Phase II uses abstractive ones. The abstractive summarizers depend upon extractive summarizer's outputs for their fine-tuning (supervision). The strategy is useful in a production environment which requires the summarization of a very large number of chat transcripts but where there is a paucity of

manually generated reference (annotated) summaries from which the abstractive summarizers can learn in Phase II and with which we can compare our results. Figure 1 shows the two Phases of the proposed hybrid summarization strategy.

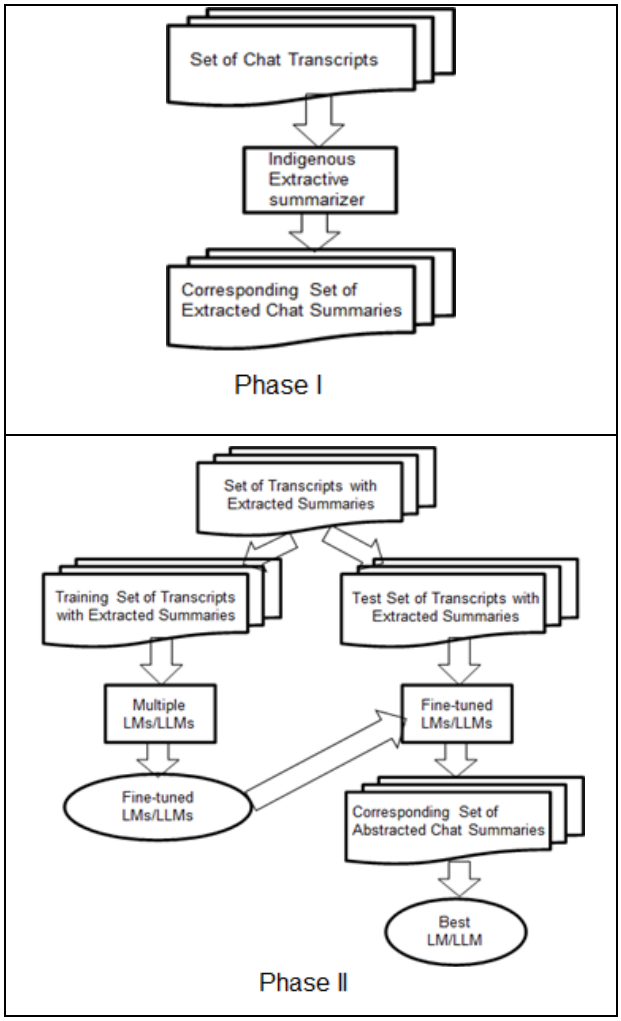

Figure 1: Hybrid Summarization.

5 Phase I: Sample Generation through Extractive Summarization

Phase I generates a large pool of chat summaries, through extractive summarization, which can be reused for fine-tuning supervised abstractive summarizers. This extractive summarization technique uniquely integrates *channel (speaker) separation*, *topic modeling*, and *similarity-based sentence selection* with *punctuation restoration* through a 10-step sequential procedure. It is *internally developed* based on an adaptation from (Biswas and Iakubovich, 2022).

The *punctuation restored summaries* are the outputs from this procedure. The procedure is highly parameterized. The full list of parameters to the proposed procedure includes: *Topic Model Type* (default: "None/False"), *Number of Topics* (default: 5), *Number of Dominant Topics* (default: 1), *Batch Size for Punctuation Restoration* (default: 512), *Term Extraction Method* (default: "global"), *Desired Summary Length* (default: 5), *Summary Table Name* (default: "summary_results"), *Word Similarity Threshold* (default: 0.5), *Uniqueness Threshold for Sentence Similarity* (default: 0.5). Next, we describe the key steps of this procedure.

5.1 Channel Separation

Chat transcripts include conversations/dialogs between customer and one or more agents and so the resultant summaries can often get mixed up. The separation of a transcript into customer and agent transcripts, based on channel or speaker identifier, can make each summary more coherent.

If the channel identifiers, associated with the transcripts, do not clearly identify the speakers then we can use a *pre-trained BERT Transformer* model with a *linear classifier* from *PyTorch nn* package as an additional layer, on top of BERT's 12 layers, to classify each dialog of the transcript into one of the two classes, i.e., *customer* and *agent* and then combine each type of dialogs to create customer and agent transcripts. We haven't used this with our chat transcripts as the speakers were identified.

5.2 Document Preparation

A *document* is a list of *keywords* extracted from each transcript and is used as input to the topic model. For document preparation, we have built a custom NLP preprocessing pipeline comprising of tokenization; removal of punctuation, extended stop-words and small words (length ≤ 4); regular expression matching; lowercasing; contraction mapping; bigrams and trigrams creation; lemmatization; parts of speech tagging and allowable tag selection. This has been implemented by combining modules available from four *Python* packages, namely, *re*, *spaCy*, *NLTK,* and *gensim*.

5.3 Topic Model Optimization and Optimal Model Selection

If the *topic model type* is specified at the invocation of the procedure, then we create multiple *topic models* (instances) of the desired type, for both customer and agent, using the *documents*, *corpus* and *vocabulary* from the corresponding chat transcripts, by varying the hyper-parameter (e.g.,

topic number) values within the pre-defined ranges by the pre-defined steps; compute their coherence scores and identify the topic models and associated hyper-parameter values that produce the best scores. Otherwise, by default, we perform the above-mentioned activity for all 3 different topic model types, namely, *LDA*, *LSI* and *HDP*, in parallel, and identify the topic models and associated hyper-parameter values that produce the best scores amongst topic models of all three types, through an extensive grid search over a wide range of values. We have exclusively used the *Python* based *gensim* package for this step.

5.4 Punctuation Restoration

The punctuation restoration algorithm is used in steps 2 and 8 of the aforesaid procedure. In step 2, we preprocess transcripts (customer and agent) to remove existing punctuations and then restore punctuations *partially*, i.e., restore only *periods* as delimiters, so that sentences can be separated in each transcript; while in step 8, we remove *existing periods* from each pair of customer and agent summaries, restore *partial* and *full punctuations* and postprocess them for more readable outputs.

We have used the *BertForMaskedLM* class of the *PyTorch BERT* model (*bert-base-uncased*[1]) for punctuation restoration and added an additional *linear* layer (*PyTorch nn module*) above the 12 BERT layers. The output of original BERT layers is a vector with the size of all vocabulary. The additional linear layer takes this as input and gives as output one of four classes, i.e., "O" (Other), "Comma", "Period" and "Question" for each encoded word. We retrained this modified BERT model using *TED transcripts*, consisting of two million words. This retraining with the proposed architecture is unique for punctuation restoration.

We found that the BERT model for punctuation restoration gave **30%** more accurate results than the LSTM based model. We implemented the punctuation restoration algorithm using *BERT Transformer*, *BertPunc* and *nn* packages, available from *PyTorch*.

5.5 Summary Generation through Sentence Selection

This process combines steps 5 through 7 of the main procedure, i.e., dominant topic identification, significant term selection and summary generation. First, we get the most *dominant topic(s)* from the selected topic models (for customer and agent) with the associated keywords for each of customer and agent documents for every transcript. Second, we use the keywords/terms associated with customer and agent dominant topics to extract the most significant inter-related *terms* for each transcript (document) pair using *word-based similarity analysis* and construct a string/document with them. Lastly, we generate fixed-length customer and agent summaries for every chat transcript, using *embedding-driven*, *sentence-based similarity analysis*. First, we condense each of customer and agent transcripts by identifying its most unique sentences and then we select a fixed number (user-specified) of most relevant sentences from the condensed transcripts that are *most similar* to the string/document constructed at the previous step.

For *term-based similarity analysis*, we have calculated *cosine similarity* between *GloVe* encoded word vectors (300 dimensions) using *spaCy*'s *en_vectors_web_lg;* while for *sentence-based similarity analysis and summary generation*, we have used the *Universal Sentence Encoder (USE)* from *tensorflow-hub,* along with the *Python* based *pandas* and *numpy* packages.

5.6 Summarization Evaluation

We have determined the effectiveness of the summarizer by measuring both the *goodness* (*quality*) of summarization and the *correctness* (*accuracy*) of the punctuation restoration reflecting the content and readability of the summaries.

For the goodness/*quality* of the information content of the generated summaries, the metrics *BLEU* [Bilingual Evaluation Understudy] (Papineni et al., 2002) and *ROUGE* [Recall-Oriented Understudy for Gisting Evaluation] (Lin, 2004) scores can be used as measurements. We have computed the *BLEU and ROUGE-l* scores using the *Python* packages *NLTK* (*nltk.translate.BLEU_score*) and *ROUGE* (*rouge*).

[1] https://huggingface.co/google-bert/bert-base-uncased

For the *correctness/accuracy* of the punctuation restoration, the *accuracy_score* function from python's *sklearn.metrics* package can measure the *punctuation-restoration-accuracy* (Biswas and Iakubovich, 2022), as the number of matches of punctuation symbols (periods) between the original/extracted text (transcript/summary) and the punctuated text (transcript/summary), expressed as a percentage.

6 Phase II: Abstractive Summarization through Transfer Learning

Phase II involves fine-tuning pre-trained *Large Language Model* (*LM/LLM*) driven, transformer based *abstractive summarizers* on the extractive summaries, obtained from Phase I, and then using them in generating summaries from unseen chat transcripts, via *transfer learning*, to find the most effective *fine-tuned* summarizer for potential production deployment. We have fine-tuned four pre-trained LM/LLM driven transformers, i.e., **T5** (*t5-small* [2]), **BART** (*bart-large-xsum* [3]), **Longformer2Roberta** (*longformer2roberta-cnn_dailymail-fp16* [4]), and **DialogLED** (*DialogLED-large-5120*[5]) for this purpose. **T5** (Raffel et al., 2019), an encoder-decoder model, was pre-trained on Common Crawl and encodes at-most 512 tokens. **BART** is a denoising transformer encoder-decoder (seq2seq) model with a bidirectional (BERT-like) encoder and an autoregressive (GPT-like) decoder, fine-tuned on the Extreme Summarization (XSum) dataset. It encodes up to 512 tokens. **Longformer2Roberta** is an encoder-decoder model, where the encoder is an *allenai/longformer-base-4096* model and the decoder is a *roberta-base* model, fine-tuned on the CNN/DailyMail dataset. It can handle up to 4096 tokens. **DialogLED** is a pre-trained model for long dialog understanding and summarization. It builds on the Longformer-Encoder-Decoder architecture and uses window-based denoising as the pre-training task on a large amount of long dialog data, encoding up to 5120 tokens. We selected **T5** as it was then the state-of-the-art, **BART** as it was then commonly used for dialog summarization, **Longformer2Roberta** as some chat transcripts were long documents and **DialogLED** as it was designed to improve dialog summarization.

7 Performance Evaluation

Effectiveness (quality of summaries), efficiency (summarization/fine-tuning time), flexibility and performance comparisons with/among open-source, off-the-shelf summarizers are some of the considerations that helped us evaluate the performances of our strategy for chat transcript summarization.

7.1 Experimental Setup

We set up a Spark cluster, consisting of a driver node and dynamically allocated, multiple executor nodes for data collection, preprocessing and summarization. The NVIDIA CUDA Deep Neural Network (cuDNN v7.6) accelerated our training process for punctuation restoration. We retrained, fine-tuned, and tested the transformer models on NVIDIA Tesla A100-SXM4-40GB GPU based nodes, using anywhere between 1 to 4 GPUs.

In Phase I, we tested our extractive summarizer on a ***dataset*** consisting of **160,000** chat transcripts, covering a wide range of issues including *billing, refunds, upgrades, service, outage, maintenance,* etc. The average and maximum lengths of the full chat and the constituent customer and agent transcripts were (314, 7295), (92, 4225) and (222, 4064) words respectively. We compared the performances of our summarizer with those from another very popular, open-source extractive summarizer, namely, *BERT Extractive Summarizer*[6] using three pre-trained transformer models: **BERT** (*bert-base-uncased*[l]) [encoder], **GPT-2** (*gpt2-medium*[7]) [decoder], and **XLNet** (*xlnet-base-cased*[8]) [decoder]. We chose these three models as they could summarize well without

[2] https://huggingface.co/google-t5/t5-small

[3] https://huggingface.co/facebook/bart-large-xsum

[4] https://huggingface.co/patrickvonplaten/longformer2roberta-cnn_dailymail-fp16

[5] https://huggingface.co/MingZhong/DialogLED-large-5120

[6] https://pypi.org/project/bert-extractive-summarizer/

[7] https://huggingface.co/openai-community/gpt2-medium

[8] https://huggingface.co/xlnet/xlnet-base-cased

fine-tuning. *BERT Extractive Summarizer* generated summaries using the *period-restored* chat, customer, and agent transcripts from step 2 of the proposed extractive summarization procedure. Its *ratio* parameter was adjusted, using the number of words in the transcript, to ensure that its summaries were of comparable (shorter) lengths. The transcripts were summarized both *with* and *without channel separation (full chat),* as part of an ***ablation study***.

In Phase II, the **160K** transcripts (customer & agent) with their corresponding extractive summaries (from Phase I) were split into 3 sets, i.e., *train*, *test* and *hold-out*, with **150K**, **5K** and **5K** samples respectively. We fine-tuned the pre-trained **T5**, **BART**, **Longformer2Roberta** and **DialogLED** models on train and test sets for abstractive summarization and validated their summaries on the hold-out set. The hyper-parameters of the models, e.g., *ecncoder_length*, *decoder_length, batch_size, num_beams*, *learning_rate*, *weight_decay*, *num_train_epochs, fp16,* etc., were tuned to generate better summaries.

The open-source summarizers were used with their respective pre-trained models, *tokenizers*, *configurations*, *vocabularies,* and *checkpoints*.

7.2 Manual Evaluation

The summaries generated by the proposed method are being manually validated for content and readability by our business customers. The goal is to subjectively evaluate if the summaries can be deemed generally useful for the very purposes that the transcripts were meant to be used. Feedback includes the following.

- For ~50 or so chat transcripts, our extractive summaries aptly matched manual summaries.
- The abstractive summaries were readable, generally comparable to the extractive summaries and mostly expressed the main information content of the original transcripts.
- If the chat was about one problem, then ~80% of the transcripts were capably summarized.
- Punctuations greatly improved the readability of the generated summaries.
- Our extractive summaries were more meaningful and readable than the summaries generated by their existing methods, namely, *genism summarizer*, *pytextrank*, *pysummarization auto-abstractor* for their use cases.
- The abstractive summaries didn't include opinions outside of those expressed in the extractive summaries (absence of bias).
- The abstractive summaries generated by **BART** matched the extractive summaries more than **T5**, **Longformer2Roberta** and **DialogLED**.

7.3 Automatic Evaluation

We evaluated our summarizer for *effectiveness* and *efficiency*. For measuring the *effectiveness* of our summarization and for comparing performances among extractive and abstractive summarizers, we have used the metrics *BLEU* and *ROUGE-l scores* (Sec. 5.6). We determined the *efficacy* of our *punctuation restoration* algorithm in Phase I using *punctuation-restoration-accuracy score* (Sec. 5.6).

The *efficiency* of a summarizer is important to real world applications. For Phase I, we have measured the *efficiency* of our extractive summarizer by recording the time taken by each of the 10 steps of our proposed procedure. We have also compared the *efficiency* of our *summary generation* process (Sec. 5.5) with that of the *BERT Extractive Summarizer* by comparing the *total time* taken by each to *summarize* all of chat, customer and agent transcripts in the 160K sample. For Phase II, we have compared the efficiency of the four abstractive summarizers by their *average fine-tuning times* on customer and agent transcripts.

7.4 Results and Summarizer Comparisons

Table 1 shows results from Phase I and compares the *effectiveness* & *efficiency* of the proposed summarizer for shorter summaries (~5 sentences) with those from the *BERT Extractive Summarizer* (BES) using three different pre-trained transformer models: **BERT**, **GPT-2**, and **XLNet** on the **160K** sample, using three different *evaluation metrics*. We compared all the extracted summaries with their corresponding *period-restored* original *transcripts* (step 2 of Phase I) for computing their *BLEU* and *ROUGE scores* as *we*

didn't have 160K manual reference summaries to compare them with. Hence, the scores were low as the compared texts were of unequal lengths. However, the situation was the same for all the compared summarizers and the objective was to determine the extent of overlap between the extracted summaries (in the four cases) and the original transcripts. The *BLEU and ROUGE scores*, for each type of transcript (chat/customer/agent), represent the *average* of the *BLEU* and *ROUGE* scores of all the summaries generated from the corresponding type of transcripts contained in the sample.

Table 1 shows that our own extractive summarizer generated chat, customer and agent summaries with higher *average BLEU* and *average ROUGE scores* than BES using the three pre-trained models: **BERT**, **GPT-2**, and **XLNet** in approximately $\frac{1}{5}, \frac{1}{7}, \frac{1}{7}$ of the time taken by BES in summarizing the transcripts separately in the three cases. This is because our method employed a faster, *embedding-based summarization* step (step 7) that reduced the search space for sentence selection. Thus, it establishes that our extractive summarizer is more *effective* and *efficient* than BES for chat transcripts. *So, it made sense to use our summaries for fine-tuning the abstractive summarizers in Phase II.* Table 1 further illustrates that extractive summarization with *channel separation* generated more coherent summaries than without separation (***ablation study***) and the customer summaries were the most effective. The *punctuation-restoration-accuracy* scores for chat, customer and agent summaries varied between $90-100\%$ in all cases. The proposed summarizer is highly parameterized and more flexible than *BERT Extractive Summarizer*.

Table 2 shows results from Phase II and compares the performances of the four fine-tuned abstractive summarizers, i.e., **T5**, **BART**, **Longformer2Roberta** and **DialogLED** on the hold-out set (**5K**). We compared all the customer and agent *abstracted* summaries with their corresponding *extracted summaries* from Phase I, for computing their *BLEU* and *ROUGE scores*. The scores were higher as texts were of comparable lengths. Table 2 shows that **BART** generated customer and agent abstractive summaries were *closest* to the extractive summaries with the highest *average BLEU* and *average ROUGE scores*, while taking the *least* fine-tuning time. Table 2 further confirms that **Longformer2Roberta** was more *effective* and *efficient* than **DialogLED** for our transcripts while **T5** was the least effective for chat transcript summarization. The **BART** models, fine-tuned on customer and agent transcripts, are being readied for production deployment.

ES	Chat BS	Chat RS	Cust. BS	Cust. RS	Agent BS	Agent RS	TST (secs)
IES	0.20	0.52	0.30	0.63	O.23	0.55	17,334 (~5 hours)
BES-1	0.13	0.44	0.27	0.59	0.16	0.47	85,867 (~24 hours)
BES-2	0.12	0.40	0.26	0.57	0.15	0.44	124,161 (~35 hours)
BES-3	0.12	0.41	0.26	0.58	0.15	0.45	118,199 (~33 hours)

Table 1: Metric scores for Extractive Summarizers. [ES: Extractive Summarizer, BS: BLEU Score, RS: ROUGE Score, TST: Total Summarization Time, IES: Indigenous Extractive Summarizer, BES-1: BES(BERT), BES-2: BES(GPT-2), BES-3: BERT(XLNet)]

AS	Cust. BS	Cust. RS	Agent BS	Agent RS	AFS (secs)
T5	0.41	0.56	0.58	0.73	50,242 (~14 hours)
BART	0.62	0.72	0.67	0.83	19,214 (~5.34 hours)
Longformer 2Roberta	0.47	0.66	0.61	0.77	79,086 (~22 hours)
DialogLED	0.46	0.63	0.55	0.74	114,574 (~32 hours)

Table 2: Metric scores for Abstractive Summarizers. [AS: Abstractive Summarizer, BS: BLEU Score, RS: ROUGE Score, AFS: Average Fine-tuning Time]

Next, we present another ***ablation study*** on the impacts of *fine-tuning* on the language models. The four language-model (LM/LLM) based abstractive summarizers were also used to summarize (*zero-shot*) the chat transcripts in the hold-out set without *fine-tuning* any of the models, to measure the full impact of *fine tuning* for chat summarization in our context. Table 3 shows results from our ***ablation study*** and demonstrates that **DialogLED** performed the best on *BLEU* scores, while **Longformer2Roberta** performed the best for *ROUGE* scores amongst the four *untuned* abstractive summarizers for all transcript types. Untuned **BART** was the least effective. Furthermore, comparing metric scores in Table 2 and Table 3, we can conclude that on an average

fine-tuning improved the performance of an abstractive summarizer on all chat transcripts by ~**8** times, on customer transcripts by ~**5** times, and on agent transcripts by ~**11** times. **BART** showed the most improvement in fine-tuning on our chat transcripts.

AS	Cust. BS	Cust. RS	Agent BS	Agent RS
T5	0.07	0.28	0.05	0.21
BART	0.03	0.21	0.01	0.16
Longformer2Roberta	0.09	0.38	0.08	0.33
DialogLED	0.17	0.36	0.14	0.29

Table 3: Metric scores in Ablation Study (Impacts of Fine-tuning). [AS: Abstractive Summarizer, BS: BLEU Score, RS: ROUGE Score]

7.5 Limitations

There are two limitations associated with the proposed method, one related to its *evaluation procedure* and the other related to its *capability*. Limitation related to its automated evaluation originated from not having enough manually crafted reference summaries for the 160K chat transcripts under consideration. In the absence of a full set of reference summaries, we compared the extracted summaries with the *period-restored* and *longer* original transcripts (from step 2 of our extractive procedure) for computing their corresponding *BLEU* and *ROUGE scores*. So, the scores were slightly lower. However, this was done for the summaries from the proposed method as well as for the three pre-trained language model driven *Bert Extractive Summarizers* to ensure consistency and similarity in the comparisons. Likewise, in the absence of manually generated reference summaries, the abstractive summarizers were fine-tuned on the extractive summaries, and we automatically compared the abstractive summaries with the extractive summaries using commonly used metric scores. However, this was done only after verifying through both automatic and some manual evaluations that our extractive summaries were highly readable and usable. On the other hand, one limitation of its capability is that it *doesn't repair grammatical errors* (one of the challenges associated with the chat transcripts), only reduces their numbers with fewer sentences, some postprocessing and abstractive summarization through pre-trained language models. This also explains the rationale behind the use of the two *denoising* abstractive summarizers in Phase II for abstractive summarization.

Furthermore, it may be noted here that this research was started several years back, prior to the arrival of the latest generation of prompt-based, all-purpose, decoder-transformer models, e.g., GPT-3, ChatGPT (GPT-3.5/4), Llama, Gemini, etc., which can also be quite effective for *zero-shot* text summarization (Zhang et al., 2024). Consequently, for this version, we considered and tested slightly earlier generation of LLMs/LMs, i.e., non-instruction-based encoder-decoder models, which had been previously used in the literature and are still utilized widely for text (e.g., dialog) summarization (generation), specifically for more customized use cases.

8 Conclusion

In this paper, we have presented a hybrid summarization technique to address some of the challenges associated with chat transcript summarization, prevalent in our context. We have combined *channel separation, topic modeling, sentence selection, punctuation restoration,* in *extractive summarization*, with *transfer learning* based *supervised abstractive summarization*, to generate coherent and more readable chat transcript summaries for a better understanding of the customer complaints and the agent resolutions. The proposed summarizer is the *only* hybrid one that restores *full punctuation* to the summaries. Finally, we have established the efficacy of the hybrid strategy through extensive experimentations and performance comparisons. The hybrid method is very useful for large-scale deployment of chat transcript summarization, in the absence of *manually* crafted reference (annotated) summaries for fine-tuning the abstractive summarizers.

References

Sanghwan Bae, Taeuk Kim, Jihoon Kim, Sang-goo Lee. 2019. Summary Level Training of Sentence Rewriting for Abstractive Summarization. *arXiv preprint arXiv:1909.08752v3*.

Itz Beltazi, Matthew E. Peters, Arman Cohan. 2020. Longformer: The Long-Document Transformer. *arXiv preprint arXiv:2004.05150v2*.

Pratik K. Biswas, Aleksandr Iakubovich. 2022. Extractive Summarization of Call transcripts. In *IEEE Access*, vol. 10, pages 119826-119840.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, Ray Kurzweil. 2018. Universal Sentence Encoder. *arXiv preprint arXiv:1803.11175v2*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. 2019. BERT: Pre-training of deep bi-directional transformers for language understanding. *arXiv preprint arXiv:1810.04805v2*.

Alexander Fabbri, Faiaz Rahman, Imad Rizvi, Borui Wang, Haoran Li, Yashar Mehdad, Dragomir Radev. 2021. ConvoSumm: Conversation Summarization Benchmark and Improved Abstractive Summarization with Argument Mining. In *Proceedings of the 59th Annual Meeting of the ACL and the 11th International Joint Conference on NLP (Volume 1: Long Papers)*, pages 6866-6880.

Guy Feigenblat, Chulaka Gunasekara, Benjamin Sznajder, Sachindra Joshi, David Konopnicki, Ranit Aharonov. 2021. TWEETSUMM - A Dialog Summarization Dataset for Customer Service. In *Findings of the ACL: EMNLP 2021*, pages 245-260.

Yihong Gong and Xin Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 19–25.

Karel Jezek and Josef Steinberger. 2008. Automatic Text Summarization. 2008. *FIIT STU Brarislava, Ustav Informatiky a softveroveho inzinierstva*. Vaclav Snasel (Ed.), pages 1- 12.

Karen Sparck Jones. 2007. Automatic summarizing: The state of the art. *Information Processing & Management*, vol. 43, no. 6, pages 1449–1481.

Atif Khan and Naomie Salim. 2014. A review on abstractive summarization Methods**.** *Journal of Theoretical and Applied Information Technology*, 2014, vol. 59, no. 1.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, Luke Zettlemoyer. 2019. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, & Comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of ACL Workshop*, Barcelona, Spain, pages 74-81.

Hui Lin and Vincent Ng. 2009. Abstractive summarization: a survey of the state of the art. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pages 9815-9822.

Linqing Liu, Yao Lu, Min Yang, Qiang Qu, Jia Zhu, and Hongyan Li, 2017. Generative Adversarial Network for Abstractive Text Summarization. *arXiv preprint arXiv:1711.09357v1*.

Yang Liu. 2019. Fine-tune BERT for extractive summarization. *arXiv preprint arXiv:1903.10318v2*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of EMNLP-IJCNLP*, pages 3730-3740.

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of research and development,* vol. 2, no. 2, pages 159–165.

Inderjeet Mani. 2001. Automatic Summarization. Natural language processing. John Benjamins Publishing Company.

Derek Miller. 2019. Leveraging BERT for text summarization on lectures. *arXiv preprint arXiv:1906.04165v1*.

Ramesh Nallapati, Bowen Zhou, C´ıcero Nogueira dos Santos, C¸ aglar Gulc'ehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*. Berlin, Germany, pages 280– 290.

Ani Nenkova and Kathleen McKeown. 2012. A survey of text summarization techniques. *In Mining Text Data*. Springer, pages 43–76.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for Automatic Evaluation of Machine Translation. In *Proceedings of 40th ACL*, Philadelphia, USA, pages 311-318.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv*:*1705.04304.*

Dragomir R Radev, Eduard Hovy, and Kathleen McKeown. 2002. Introduction to the special issue on summarization. *Computational linguistics,* vol. 28, no. 4, pages 399–408.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J. Liu. 2019. Exploring the Limits of Transfer Learning with a unified Text-to-Text Transformer. *arXiv preprint arXiv:1910.10683*.

Horacio Saggion and Thierry Poibeau. 2013. Automatic text summarization: Past, present and future. In *Multi-source, Multilingual Information Extraction and Summarization*. Springer, pages 3–21.

Abigail See, Peter J. Liu, Christopher D. Manning. 2017. Get to the point: summarization with pointer-

generator networks. *arXiv preprint arXiv:1704.04368v2*.

Ming-Hsiang Su, Chung-Hsien Wu, Hao-Tse Cheng. 2020. A two-stage Transformer-based Approach for Variable Length Abstractive Summarization. *IEEE/ACM TASLP*, vol. 28, pages 2061-2072.

Don Tuggener, Margot Mieskes, Jan Deriu, and Mark Cieliebak. 2021. Are We Summarizing the Right Way? A Survey of Dialogue Summarization Data Sets. In *Proceedings of the Third Workshop on New Frontiers in Summarization*, pages 107–118.

Dingding Wang, Shenghuo Zhu, Tao Li, and Yihong Gong. 2009. Multi document summarization using sentence-based topic models. In *Proceedings of the ACL-IJCNLP Conference: Short Papers*, pages 297-300.

Klaus Zechner. 1997. A Literature Survey on Information Extraction and Text Summarization. Computational Linguistics Program, Carnegie Mellon University, April 14.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B. Hashimoto. 2024. Benchmarking Large Language Models for News Summarization. *Transactions of the Association for Computational Linguistics*, vol. 12, pages 39–57.

Ming Zhong, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. DialogLM: Pre-trained Model for Long Dialogue Understanding and Summarization. *arXiv preprint arXiv:2109.02492*.