TWEETSUMM - A Dialog Summarization Dataset for Customer Service

Guy Feigenblat†, Chulaka Gunasekara, Benjamin Sznajder, Ranit Aaronov, David Konopnicki, Sachindra Joshi
IBM Research AI
{guyf@il, chulaka.gunasekara@, benjams@il}.ibm.com
{ranit.aharonov2@, davidko@il, jsachind@in}.ibm.com

Abstract

In a typical customer service chat scenario, customers contact a support center to ask for help or raise complaints, and human agents try to solve the issues. In most cases, at the end of the conversation, agents are asked to write a short summary emphasizing the problem and the proposed solution, usually for the benefit of other agents that may have to deal with the same customer or issue. The goal of the present article is advancing the automation of this task. We introduce the first large scale, high quality, customer care dialog summarization dataset with close to 6500 human annotated summaries. The data is based on real-world customer support dialogs and includes both extractive and abstractive summaries. We also introduce a new unsupervised, extractive summarization method specific to dialogs.

1 Introduction

Text summarization is the task of creating a short version of a long text, retaining the most important or relevant information. In NLP, there are two types of summarization tasks- (1) Extractive summarization, in which segments from the original text are selected to form a summary and (2) Abstractive summarization, in which new natural language expressions are generated for summarizing the text. The past few years have witnessed a tremendous progress in creating both kinds of summaries using seq2seq models. However, these works have largely focused on documents such as news and scientific publications (Lin and Ng, 2019).

In this paper, we focus on summarizing conversational data between customers and human support agents. In many enterprises, once an agent is done with handling a customer request, she is required to create a short summary of the conversation for record keeping purposes. At times, an ongoing conversation may also need to be transferred to another agent or escalated to a supervisor. This also requires creating a short summary of the conversation so far, as to provide the right context to the next handling agent.

Our main contribution is the release of TWEETSUMM, a dataset focused on summarization of dialogs, which represents the rich domain of Twitter customer care conversations 1. The dataset contains close to 6500 extractive and abstractive summaries generated by human annotators from 1100 dialogs. This is the first dataset released to the research community, which focuses on real dialogs, as opposed to previous works focusing on meeting conversations (McCowan et al., 2005), general chitchat summarization (Gliwa et al., 2019), or topic descriptions of interviews (Zhu et al., 2021). Furthermore, the fact that each dialog was annotated by 3 different crowd-workers, resulting in an overall of 6 summaries for each dialog, provides diversity of summaries. We performed quality control and assessment to remove erroneous summaries, and to ensure that the collected TWEETSUMM summaries are of a high quality. We evaluate several summarization baselines and further provide a novel unsupervised extractive summarization algorithm, referred to as NRP Summ which outperforms other unsupervised baselines for extractive summarization. Figure 2 shows an example of a TWEETSUMM dialog along with a human-generated abstractive summary and two machine-generated summaries - abstractive and extractive summaries. We propose that the dataset quality and scale, is suitable for developing future models for the dialog summarization task. We hope that releasing TWEETSUMM for the community will foster further research.

1https://github.com/guyfe/Tweetsumm

*With equal contribution
†Current address: guy@piiano.com
‡Current address: david.konopnicki@booking.com
2.1 Dialog Reconstruction Method

The data is delivered via a CSV file where each record contains the following fields: text - the anonymized text of the Tweet, tweet_id - unique anonymized Tweet ID, author_id - unique anonymized author ID, inbound - whether the Tweet is to or from a company, response_tweet_id - IDs of Tweets that are responses to this Tweet, in_response_to_tweet_id - ID of the Tweet this Tweet is in response to, and created_at - date and time the Tweet was sent.

In order to reconstruct dialogs from Tweets, we traversed the CSV data recursively using the in_response_to_tweet_id field. At the end of this process, each dialog is a sorted list of Tweets and their metadata fields. In case several Tweets are posted as response to the same Tweet, they are sorted by their created_at timesamp. This often happens when a message exceeds the length limit for a single Tweet, and has to be split.

2.2 Summaries Generation

Each annotator was asked to generate one extractive and one abstractive summary for a single dialog at a time. When generating the extractive summary, the annotators were instructed to highlight the most salient sentences in the dialog. For the abstractive summaries, they were instructed to write a summary that contains one sentence summarizing what the customer conveyed and a second sentence summarizing what the agent responded. See the supplementary material for a detailed description of the instructions provided to annotators before starting the task. We collected 3 annotations per dialog, such that overall we obtained \( \approx 6600 \) summaries: \( \approx 3300 \) extractive summaries, termed hereafter the extractive dataset and \( \approx 3300 \) abstractive summaries, termed hereafter the abstractive dataset.

As explained in the next section, some summaries were discarded following quality control, and for some dialogs, a second round of summaries collection was done. Overall, TWEETSUMM contains 3056 extractive and 3327 abstractive summaries.
2.3 Quality Control and Assessment

2.3.1 Quality Control

To guarantee a high quality level of annotations, multiple measures were taken in advance. We only recruited as crowd-workers, members of an Expert Business Partner channel, who are fluent English speakers. Before an annotator was approved for the task, he or she had to pass a quality control test by annotating 10 dialogs with an acceptable high quality. The quality of those summaries was checked manually. Out of 25 annotators who participated in the test only 10 were approved for the task.

Following completion of the task, several heuristics were applied to identify and discard bad extractive summaries, and statistics were kept on annotators to identify those, if any, that produced erroneous summaries with high frequency. The applied heuristics included removing summaries containing only one sentence, summaries containing only one side (Customer-only or Agent-only), or summaries starting by an Agent turn. We remove summaries starting by an agent turn since tweeter dialogs begin by a customer raising an issue, and hence the summary is expected to begin with a customer turn. By these cleansing steps, we removed from our dataset 286 extractive summaries. None of the annotators exhibited a high frequency of such bad summaries, supporting the assumption that these errors are due to technical annotation problems, such as erroneously pressing submit prematurely, rather than an annotator performing poorly on the task in general.

To further assure the quality of the summaries, we computed on each document and for each annotator the percentage of his selected sentences which were also selected by one of the other annotators. A classical Jacquard score would result in irrelevant low-scores if one of the other annotators selected a large number of sentences, and, thus, we used a slightly adapted version $J = \frac{|A \cap B|}{|A|}$ which punishes $A$ if he selected a less concise summary. No annotator got an extreme low score and the average scores of the annotators range from 50% to 68%. For extra safety, we manually checked the summaries with low $J$ scores and found that they do not appear to be unequivocally erroneous. Rather, the difference in the selection of the sentences was due to similar sentences in the original dialog and to the inherent subjectivity of the task, which is also consistent with previous research (Daume III and Marcu, 2005).

Following completion of the task, several heuristics were applied to identify and discard bad extractive summaries, and statistics were kept on annotators to identify those, if any, that produced erroneous summaries with high frequency. The applied heuristics included removing summaries containing only one sentence, summaries containing only one side (Customer-only or Agent-only), or summaries starting by an Agent turn. We remove summaries starting by an agent turn since tweeter dialogs begin by a customer raising an issue, and hence the summary is expected to begin with a customer turn. By these cleansing steps, we removed from our dataset 286 extractive summaries. None of the annotators exhibited a high frequency of such bad summaries, supporting the assumption that these errors are due to technical annotation problems, such as erroneously pressing submit prematurely, rather than an annotator performing poorly on the task in general.

To further assure the quality of the summaries, we computed on each document and for each annotator the percentage of his selected sentences which were also selected by one of the other annotators. A classical Jacquard score would result in irrelevant low-scores if one of the other annotators selected a large number of sentences, and, thus, we used a slightly adapted version $J = \frac{|A \cap B|}{|A|}$ which punishes $A$ if he selected a less concise summary. No annotator got an extreme low score and the average scores of the annotators range from 50% to 68%. For extra safety, we manually checked the summaries with low $J$ scores and found that they do not appear to be unequivocally erroneous. Rather, the difference in the selection of the sentences was due to similar sentences in the original dialog and to the inherent subjectivity of the task, which is also consistent with previous research (Daume III and Marcu, 2005).

Following completion of the task, several heuristics were applied to identify and discard bad extractive summaries, and statistics were kept on annotators to identify those, if any, that produced erroneous summaries with high frequency. The applied heuristics included removing summaries containing only one sentence, summaries containing only one side (Customer-only or Agent-only), or summaries starting by an Agent turn. We remove summaries starting by an agent turn since tweeter dialogs begin by a customer raising an issue, and hence the summary is expected to begin with a customer turn. By these cleansing steps, we removed from our dataset 286 extractive summaries. None of the annotators exhibited a high frequency of such bad summaries, supporting the assumption that these errors are due to technical annotation problems, such as erroneously pressing submit prematurely, rather than an annotator performing poorly on the task in general.

To further assure the quality of the summaries, we computed on each document and for each annotator the percentage of his selected sentences which were also selected by one of the other annotators. A classical Jacquard score would result in irrelevant low-scores if one of the other annotators selected a large number of sentences, and, thus, we used a slightly adapted version $J = \frac{|A \cap B|}{|A|}$ which punishes $A$ if he selected a less concise summary. No annotator got an extreme low score and the average scores of the annotators range from 50% to 68%. For extra safety, we manually checked the summaries with low $J$ scores and found that they do not appear to be unequivocally erroneous. Rather, the difference in the selection of the sentences was due to similar sentences in the original dialog and to the inherent subjectivity of the task, which is also consistent with previous research (Daume III and Marcu, 2005).

In addition, we looked for cases where annotators used a repeating, or closely-repeating, text for abstractive summaries of different dialogs. We have identified only 9 such abstractive summaries, which were discarded from the dataset.

### Table 1: Results of the Quality Assessment

<table>
<thead>
<tr>
<th>Summary Type</th>
<th>Question</th>
<th>Average score</th>
</tr>
</thead>
<tbody>
<tr>
<td>extractive</td>
<td>Provide your rating as to the overall coverage of the summary, based on how well it represents important information from the dialog</td>
<td>4.03 (±0.77)</td>
</tr>
<tr>
<td>extractive</td>
<td>Provide your rating as to the readability of the summary. Please consider fluency, grammatical correctness, and coherence</td>
<td>4.22 (±0.61)</td>
</tr>
<tr>
<td>abstractive</td>
<td>Provide your rating as to the overall coverage of the summary, based on how well it represents important information from the dialog</td>
<td>3.96 (±0.84)</td>
</tr>
<tr>
<td>abstractive</td>
<td>Provide your rating as to the readability of the summary. Please consider fluency, grammatical correctness, and coherence</td>
<td>4.03 (±0.84)</td>
</tr>
</tbody>
</table>

We thus conclude, based on our quality control and assessment, that the TweetSumm dataset contains high quality summaries generated by high quality annotators.
2.4 Dataset Analysis

Table 2 details the average length of the dialogs in TWEETSUMM, including the average lengths of the customer and agent utterances. The average length of the summaries is reported in Table 3. Comparing the dialog lengths to the summaries lengths indicates the average compression rate of the summaries. For instance, on average, the abstractive summaries compression rate is 85% (i.e. the number of tokens is reduced by 85%), while the extractive summaries compression rate is 70%. The number of customer and agent sentences selected in the extractive summaries were relatively equally distributed with 7445 customer sentences and 7844 agent sentences in total.

<table>
<thead>
<tr>
<th></th>
<th>Full dialog</th>
<th>Customer utterances</th>
<th>Agent utterances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utterances</td>
<td>10.17±(±2.31)</td>
<td>7.49±(±1.84)</td>
<td>4.69±(±1.09)</td>
</tr>
<tr>
<td>Sentences</td>
<td>22(±5.56)</td>
<td>10.23(±4.83)</td>
<td>11.75(±4.44)</td>
</tr>
<tr>
<td>Tokens</td>
<td>245.01(±79.16)</td>
<td>125.61(±63.94)</td>
<td>119.40(±46.73)</td>
</tr>
</tbody>
</table>

Table 2: Average lengths of dialogs

Table 3: Average lengths (in # tokens) of summaries

<table>
<thead>
<tr>
<th></th>
<th>Overall</th>
<th>Customer</th>
<th>Agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstractive</td>
<td>36.41(±12.97)</td>
<td>16.85(±7.23)</td>
<td>19.56(±8.27)</td>
</tr>
<tr>
<td>Extractive</td>
<td>73.57(±28.80)</td>
<td>35.59(±21.3)</td>
<td>35.80(±18.67)</td>
</tr>
</tbody>
</table>

For each, we create a set of \( k + 1 \) (\( k=5 \) in this study) triples: one triple containing the correct response \( r \) \((\text{label=1})\), and \( k \) triples containing incorrect responses randomly sampled from the dataset \((\text{label=0})\). The dialog context \( C \) and a candidate response \( c_i \) are fed together to BERT as a sequence \(([CLS] C [SEP] c_i [SEP])\). The hidden state of the [CLS] token was used as the representation of the pair. Training is done using positive and negative examples with cross-entropy loss. A model trained on the NRP task associates a probability \( p_r \) for the response \( r \), given the context \( C \). We trained two NRP models, (1) a model predicting the next response given the prior sentences \((NRP-FW)\), and (2) a model predicting the prior utterance given subsequent utterances \((NRP-BW)\).

### Salient sentence identification

The intuition behind this approach is that the removal of the critical sentences from a dialog context will entail a larger drop in probability in predicting a subsequent and prior responses. We follow the hypothesis that the critical sentences for the NRP task will also be salient sentences for the summary. The sentence removal occurs in two steps. In the initial step, we feed the entire context to the NRP model and identify the probability of predicting the next (or prior) sentence. In the next step, we remove one sentence at a time from the context, and input the new context to the NRP model and identify the probability of predicting the same next (or prior) utterance. Then, we assign the drop in probability as a score to the removed sentence.

To identify the salient sentences in predicting...
the next response, we remove one sentence at a
time from the dialog context \((C\setminus s_i)\) and use that
as the input to a trained \(NRP-FW\) model and iden-
tify the probability \((p_r^f)^w\) for the corresponding
response \((c_r)\). Then, we assign the drop in proba-
bility \((p_r−p_r^f)^w\) as a score to the removed sentence
\(s_i\) in the context. We follow the same process to
identify the drop in probability in predicting the
prior sentence, given the same dialog context and
masked sentence (using \(NRP-BW\) model), and as-
sign that as another score for the masked sentence.
The averaged score for each sentence is used during
salient sentence identification. For the evaluation,
we use the top two customer sentences and the two
top agent sentences as the extractive summary of
the dialog.

4 Experiments and Results

We aim to confirm that TWEETSUMM is suitable
as a ground-truth dataset for the dialog summa-
ризacji задачи. To this end, we apply and analyze
several baseline summarization models as well as
\(NRP Summ\), to the dataset, as detailed below. We
randomly split the dialogs and their associated sum-
maries into three sets: 80% for the training set, 10%
for the validation and the rest 10%, for the test set.

4.1 Baselines

The baselines evaluated as part of this study are:
Random (extractive) - Two random sentences from
the agent utterances and two from the customer ut-
terances.

LEAD-4 (extractive) - The first two sentences from
the agent utterances and the first two from the cu-
tomer utterances. This approach is considered a
very competitive baseline (see (Kryscinski et al.,
2019) when considering news summarization).

LexRank (extractive) - This unsupervised summa-
rizier (Erkan and Radev, 2004) casts the summariza-
tion problem into a fully connected graph, in which
nodes represent sentences and edges represent sim-
ilarity between two sentences. Pair-wise similarity
is measured over the bag-of-words representation
of the two sentences. Then, \(PowerMethod\) is ap-
plied on the graph, yielding a centrality score for
each sentence. We take the top two central cus-
tomer and agent sentences \((2+2)\).

Cross Entropy Summarizer (extractive)- CES is
an unsupervised, extractive summarizer (Roitman
et al., 2020; Feigenblat et al., 2017), which consid-
ers the summarization problem as a multi-criteria
optimization over the sentences space, where sev-
eral summary quality objectives are considered.

The aim is to select a subset of sentences optimiz-
ing these quality objectives. The selection runs in
an iterative fashion: in each iteration, a subset of
sentences is sampled over a learned distribution
and evaluated against quality objectives. We intro-
duced some minor tuning to the original algorithm,
to suit dialog summarization. First, query quality
objectives were removed since we focus on generic
summarization. Then, since dialog sentences tend
to be relatively short, when measuring the cover-
age objective, each sentence was expanded with the
two most similar sentences, using Bhattacharyya
similarity. Finally, Lex-Rank centrality scores were
used as an additional quality objective, by averag-
ing the centrality scores of sentences in a sample.

PreSumm (extractive/abstractive) - This model
(Liu and Lapata, 2019b) applies BERT (Devlin
et al., 2019) for text summarization in both ex-
tractive and abstractive settings. In the extractive
setting, PreSumm treats the summarization task as
a sentence classification problem: a neural encoder
creates sentence representations and a classifier pre-
dicts which sentences should be selected for the
summary. We used a pre-trained model and fine-
tuned the model using the TWEETSUMM. In the
abstractive setting, the model uses the same en-
coder as the extractive model while the decoder is
a 6-layered Transformer initialized randomly.

BART (abstractive) - A denoising autoencoder
(Lewis et al., 2019) that uses the seq2seq trans-
former architecture. It consists of two parts: an
encoder and a decoder. The encoder is a bidirec-
tional encoder which corresponds to the structure
of BERT, and the decoder is an auto-regressive
decoder following the settings of GPT (Radford
et al., 2019). We use a lightweight variant of BART
(coined DistilBART) that is fine-tuned on the XSum
 task (Narayan et al., 2018b). We further fine-tuned
the model using the TWEETSUMM. Different vari-
ants of the BART model that were evaluated are dis-
cussed in the results section. The hyper-parameters
are described in the supplemental material.

4.2 Automatic Evaluation

We first use automatic measures to evaluate the
summaries generated by the models described
above, using the reference summaries of TWEET-
SUMM. We measured summarization quality using

\[ \frac{1}{N} \sum_{i=1}^{N} \text{score}(S_i) \]

where \(S_i\) is the generated summary and \(\text{score}(S_i)\)
is the automatic score assigned to it.
Table 4: ROUGE F-Measure evaluation on the test set, supervised baselines are marked with †.

<table>
<thead>
<tr>
<th>Length Limit</th>
<th>Method Name</th>
<th>R-1</th>
<th>R-2</th>
<th>R-5/4</th>
<th>R-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>35 tokens</td>
<td>Random</td>
<td>63.91</td>
<td>63.99</td>
<td>20.044</td>
<td>20.044</td>
</tr>
<tr>
<td></td>
<td>Lead</td>
<td>63.91</td>
<td>63.99</td>
<td>20.044</td>
<td>20.044</td>
</tr>
<tr>
<td></td>
<td>LexRank</td>
<td>27.66</td>
<td>10.988</td>
<td>11.698</td>
<td>24.369</td>
</tr>
<tr>
<td></td>
<td>CES</td>
<td>27.66</td>
<td>10.988</td>
<td>11.698</td>
<td>24.369</td>
</tr>
<tr>
<td></td>
<td>NRP Summ</td>
<td>30.197</td>
<td>12.219</td>
<td>13.911</td>
<td>27.111</td>
</tr>
<tr>
<td></td>
<td>BART - without fine-tuning</td>
<td>20.356</td>
<td>14.110</td>
<td>13.813</td>
<td>26.536</td>
</tr>
<tr>
<td></td>
<td>PreSumm extractive</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
<tr>
<td></td>
<td>PreSumm abstractive</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
<tr>
<td></td>
<td>CES</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
<tr>
<td></td>
<td>NRP Summ</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
<tr>
<td></td>
<td>Lead</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
<tr>
<td>70 tokens</td>
<td>Random</td>
<td>63.91</td>
<td>63.99</td>
<td>20.044</td>
<td>20.044</td>
</tr>
<tr>
<td></td>
<td>Lead</td>
<td>63.91</td>
<td>63.99</td>
<td>20.044</td>
<td>20.044</td>
</tr>
<tr>
<td></td>
<td>LexRank</td>
<td>30.197</td>
<td>12.219</td>
<td>13.911</td>
<td>27.111</td>
</tr>
<tr>
<td></td>
<td>CES</td>
<td>30.197</td>
<td>12.219</td>
<td>13.911</td>
<td>27.111</td>
</tr>
<tr>
<td></td>
<td>NRP Summ</td>
<td>30.197</td>
<td>12.219</td>
<td>13.911</td>
<td>27.111</td>
</tr>
<tr>
<td></td>
<td>BART - without fine-tuning</td>
<td>20.356</td>
<td>14.110</td>
<td>13.813</td>
<td>26.536</td>
</tr>
<tr>
<td></td>
<td>PreSumm extractive</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
<tr>
<td></td>
<td>PreSumm abstractive</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
<tr>
<td></td>
<td>CES</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
<tr>
<td></td>
<td>NRP Summ</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
<tr>
<td></td>
<td>Lead</td>
<td>65.66</td>
<td>56.628</td>
<td>54.327</td>
<td>64.943</td>
</tr>
</tbody>
</table>

4.2.1 TweetSumm Abstractive Dataset

Quality of extractive summarization models: We start by analyzing how well extractive summarization models perform on the abstractive reference summaries. As shown in Table 4, we note that in most cases, except 70 tokens summary, NRP Summ outperforms other unsupervised, extractive baselines. Interestingly, the performance of the simple Lead-4 baseline is not far from that of the more complex unsupervised baselines. For instance, considering the 70 tokens results of the abstractive dataset, LexRank outperforms Lead-4 by only 4%-8%. This is backed up by the statistics we report in section 2.4, namely that salient content conveyed by the customer appears at the beginning of the dialog. To rule out any potential overfitting, we also present results of the unsupervised, extractive, summarizers against the validation set. Table 5 shows a similar trend: in most cases, NRP Summ outperforms other models.

Quality of abstractive summarization models: We analyze three variants of the BART model: (1) BART with no fine-tuning on TweetSumm (BART-without-fine-tuning). (2) BART fine-tuned on TweetSumm (BART-without-ext), (3) BART fine-tuned on TweetSumm with the extractive summary provided as input in addition to the dialog (BART-with-ext). For training the BART-with-ext, the ground truth extractive summaries were appended to the dialog (with a dedicated separator). For validation and testing, the extractive summaries generated by the NRP Summ model were used. All BART models were pre-trained on the XSum summarization dataset (Narayan et al., 2018a) (see the specific system models settings in the supplemental material). As described in Table 4, the BART models fine-tuned on TweetSumm obtain the best results by far, compared to all other models. BART-without-fine-tuning model performs poorly, compared to all the other models. From this analysis we learn that, pre-training on the general summarization task is not sufficient, fine-tuning is required to help the model learn the specifics of the dialog summarization task. Interestingly, BART-without-ext outperforms BART-without-ext, suggesting that the extractive summary helps the model to attend to salient content. Although the PreSumm model was also similarly fine-tuned on TweetSumm, its performance is inferior to BART.
4.2.2 TweetSumm Extractive Dataset

Here we focus on evaluating the extractive summarization models on the extractive dataset. We first note that the average length of ground truth extractive summaries in TweetSumm is 4 sentences out of 22 sentences, on average, in a dialog. The lower compression rate of the extractive summaries compared to the abstractive summaries leads to higher ROUGE scores of the extractive summaries. The NRP Summ model outperforms all unsupervised methods, while the supervised PreSumm extractive model outperforms all other models.

4.3 Human Evaluation

We conducted two human evaluation studies to assess the quality of the summarization models. The first focuses on the Informativeness and Saliency of the summaries generated by the models. Following (Liu and Lapata, 2019c, a), we used the QA paradigm to test whether the summarization models retain key information. We chose to evaluate the two abstractive models BART-without ext and PreSumm-abs and four extractive models - NRP Summ, CES, PreSumm-ext and LEAD (limited to 4 sentences). We randomly selected 20 dialogs and recruited 4 NLP expert annotators for the task. One was asked to create a set of questions based on the three ground truth abstractive summaries from TweetSumm, and the other three were asked to read the generated summaries and answer the questions. Using the abstractive rather than the extractive summaries allows the questions to focus on the most salient information, since the extractive summaries are constrained by having a limit of 4 sentences selected as-is from the dialog. For each dialog, 4–10 yes/no questions regarding the information included in the summary (e.g. “Does the summary specify that...”), were created by the human annotator. Following (Nenkova and Passonneau, 2004), we assigned each question a weight, \( w_j \), which is the ratio of ground-truth summaries containing an answer for question \( j \). Clearly, important information should be included in several human summaries. Then, the other three annotators, \( i \in \{1, 2, 3\} \) were given the set of questions and one summary at a time (without knowing which model generated the summary), and were asked to indicate whether the summary contained an answer to the question. Denote the indicator \( I_{ij} \) to be 1 if annotator \( i \) determined that the summary contained an answer to question \( j \), and 0 otherwise. The score of a summary generated by a model per dialog \( d \) is calculated as

\[
S_d = \frac{100}{K_d} \sum_{j=1}^{K_d} w_j \sum_{i=1}^{3} I_{ij}
\]

where \( K_d \) is the number of questions given \( d \). The highest score a summary can get is 100 which occurs when all annotators agreed that the summary includes the information in all questions. Refer to the supplemental material for examples of questions that were created as part of this evaluation.

Table 6 reports the evaluation results, when calculating the summary scores separately for questions pertaining to the agent and customer utterances. The Lead-4 baseline outperforms other methods for summarizing customer utterances, which is expected as remarked in sub-section 4.2.1. In this case, the simple baseline is hard to beat. However, for summarizing agent utterances, the more advanced models are better, but even the supervised PreSumm and BART models leave much room for improvement.

Following (Liu and Lapata, 2019a), we further assess the quality of the summaries along the two dimensions of Readability and Informativeness. We chose to evaluate only the abstractive models (BART-without ext and PreSumm) since a high level of Readability is not expected with extractive summaries. The annotators were asked to indicate which summary is better with respect to their Readability and Informativeness, without knowing which system was used to generate which summary. In more than 90% of the cases BART outperforms PreSumm on both dimensions, consistent with the results in Table 6.

4.4 Further Analysis of BART summaries

In section 4.2.1 we showed that fine tuning BART on TweetSumm significantly improves the summaries compared to using BART with no fine tuning. Here we examine, whether using TweetSumm for fine tuning improves BART’s ability to learn an important characteristic of dialog summarization, namely, that a summary should convey text from both speakers (agent and customer). We consider three variants of BART: (1) BART fine tuned on TweetSumm, (2) BART fine tuned as in (1) for which additional speaker tags (agent or customer) were added during fine tuning, (3) original BART variant, with no fine-tuning on TweetSumm. We generate summaries for each dialog in the test set using each of the aforementioned variants (1)-(3). Following (Nallapati et al., 2017),
works different strategies are employed to identify existing texts that can be used as reference summaries. Summarization of scientific articles has also been studied as in (Yasunaga et al., 2019), treating abstracts as well as sentences describing another paper, as potential reference summaries.

Data Driven Dialog Systems: Many aspects of data driven dialog systems have undergone a revolution in recent years with the advent of ever more powerful techniques based on deep learning (Serban et al., 2016; Henderson et al., 2019; Zhang et al., 2019; Wu et al., 2020). Most of the available dialog datasets support dialog tasks such as next response prediction (Kadlec et al., 2015; Bordes et al., 2016; Byrne et al., 2019), conversational question answering (Reddy et al., 2019; Choi et al., 2018; Saeidi et al., 2018) and dialog state tracking (Budzianowski et al., 2018; Rastogi et al., 2019).

Dialog Summarization Datasets- On the other hand, summarization of two-party dialogs is relatively unexplored due to the lack of suitable large scale benchmark data. Most of the previous works on abstractive dialog summarization (Banerjee et al., 2015; Mehdad et al., 2014; Goo and Chen, 2018; Li et al., 2019) focus on the AMI meeting corpus dataset (McCowan et al., 2005). This dataset has multiple deficiencies including, its size (only 141 summaries are available), and the quality of the ground truth summaries, since the meeting description is treated as the summary. The Argumentative Dialog Summary Corpus (Misra et al., 2015), a small dataset of 45 dialogs, is based on political debates from the Internet Argument Corpus (Walker et al., 2012) where summaries are constructed by crowd-workers. More recently, CRD3 (Ramesh Kumar and Bailey, 2020) was introduced, a spoken conversation dataset that consists of 159 conversations and summaries. The SAMSum dialog corpus (Gliwa et al., 2019) contains over 16k chat conversations with manually annotated abstractive summaries. However, this dataset contains role-playing open domain, chitchat dialogs, and does not provide ground truth for extractive summarization. In contrast, TWEETSUMM involves different summarization challenges, e.g., identifying problems and provided solutions. (Yuan and Yu, 2019) studied the problem of abstractive dialog summarization using a dataset constructed from the MultiWOZ-2.0 dataset (Budzianowski et al.,

Table 6: System scores based on questions answered

<table>
<thead>
<tr>
<th>Model</th>
<th>Type</th>
<th>Customer</th>
<th>Agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEAD</td>
<td>ext</td>
<td>77.9</td>
<td>39.2</td>
</tr>
<tr>
<td>CES</td>
<td>ext</td>
<td>69.6</td>
<td>49.9</td>
</tr>
<tr>
<td>NRP Summ</td>
<td>ext</td>
<td>71.3</td>
<td>40.8</td>
</tr>
<tr>
<td>PrecSumm</td>
<td>ext</td>
<td>74.3</td>
<td>51.2</td>
</tr>
<tr>
<td>PrecSumm†</td>
<td>abs</td>
<td>16.0</td>
<td>12.5</td>
</tr>
<tr>
<td>BART(without-ext)</td>
<td>abs</td>
<td>58.5</td>
<td>31.7</td>
</tr>
</tbody>
</table>
This dataset considers the instructions provided to crowd-workers as part of the Wizard-of-OZ setting as the ground truth summary. Hence, the dataset does not contain “real” summary annotations for dialogs. (Liu et al., 2019) worked on the problem of automatic summary generation for customer service dialogs, but the dataset is not publicly available. Recently, MediaSum (Zhu et al., 2021) was released, suggesting the use of overview and topic descriptions as summaries of 460k interview transcripts from NPR radio channel.

6 Conclusion

In this paper, we release TweetSumm, the first open large-scale dataset focused on summarization of customer-support dialogs. We conducted automatic and human evaluation studies to ensure the high-quality of the human-generated extractive and abstractive summaries. To test the applicability of the dataset, we evaluated various baselines, as well as a new extractive summarization method, NRP Summ, and showed that while automatically generated abstractive summaries achieve high quality, there is still much room for improvement. We believe TweetSumm will help foster research in this real-world scenario, which was previously little studied due to lack of suitable datasets.

7 Ethics

We constructed TweetSumm dialogs using the publicly available Customer Support on Twitter dataset (www.kaggle.com/thoughtvector/customer-support-on-twitter). The summaries generation task was executed on Appen.com platform; we only recruited crowd-workers that are members of an Expert Business Partner channel, fluent English speakers, with a very high approved task acceptance rate. We have set the task payment, so that crowd-workers are expected to earn 9$ per hour.

References


Hui Lin and Vincent Ng. 2019. Abstractive summarization: A survey of the state of the art. In AAAI.


A TWEETSUMM Dataset - Summaries Generation

As described in the main paper, TWEETSUMM dialogs were sent for generation of summaries using crowd-sourcing on the Appen.com platform. Figure 3 shows the instructions provided to annotators working on TWEETSUMM summary generation task. Figure 4 shows how dialogs were presented to annotators as part of the annotation interface. Figure 5 shows the dialog annotation interface: annotators are asked to highlight the salient sentences (extractive summary) in the dialog. In the following sub-sections we describe in details the instructions crowd-workers received while working on this task.

A.1 Extractive Summaries

The annotators were asked to select 2 to 3 entire sentences that describe the most important messages the customer conveyed. They were asked to focus on sentences presenting a problem, complaint, or a request the customer expressed. Then, they were asked to select between 2 to 3 entire sentences representing the agent response to the customer, with focus on actual solutions and not on apologies or gratitude expressions. Clearly, the analysis of the emotional part of customer interactions is also important. However, this is associated with other NLP tasks such as sentiment analysis. The same decision was taken in (Liu et al., 2019). As a final step, the annotators were asked to go over the selected summary sentences and make sure that they represent the full dialog as much as possible. In addition, several examples of uninformative sentences, that should not appear in summaries, were given to help annotators understand the requirements better (e.g. “We’re sorry to hear that.”, “Poor customer service.”, “Hi again, we’d like to investigate this behavior.”, “I hate X company”).

A.2 Abstractive Summaries

Here, the annotators were instructed to write two sentences summarizing the whole dialog, one summarizing the customer questions/requests and the second one summarizing the agent responses. We limited ourselves to two sentences to simplify the task of the crowd-workers. In addition, having separate summary sentences allow an automated summarizer to (potentially) generate two summaries, one for the customer and one for the agent. Similarly to the extractive summarization, annotators were asked to write an informative summary, that focuses on requests, problem descriptions and solutions excluding personal opinions, insults or apologies.

B Model Training and Hyperparameter Details

In this section, we elaborate the training processes and the hyperparameters used in the supervised trained models used in this study. Each experiment was run on 2 V100 GPUs (on a single machine).

B.1 Next response prediction model for NRP Sum

As introduced in the main paper, the NRP Sum model uses a BERT based binary classifier. The code will be open-sourced in a public git page upon paper acceptance. For this task, we used the Bert-ForSequenceClassification model of HuggingFace (Wolf et al., 2019), commonly used for GLUE tasks (Wang et al., 2018). We process the dataset to construct triples of <dialog context (C), candidate (ci), label (1/0)> from each dialog context. For each C, we create a set of 10 triples: one triple containing the correct response (label=1), and 9 triples containing incorrect responses randomly sampled from the dataset (label=0). Training is done using positive and negative examples with cross-entropy loss.

The hyperparameters used for training the model are as follows:

```plaintext
model=bert-base-cased
do_lower_case=True
max_seq_length=512
per_gpu_eval_batch_size=24
per_gpu_train_batch_size=24
learning_rate=2e-5
num_train_epochs=5
adam_epsilon=1e-8
max_grad_norm=1.0
```

We trained two models with this approach, one for predicting the next response given a dialog context and, another to predict the previous sentence given the dialog context. The results of the two models on the validation set are shown in Table 1.

B.2 PreSumm model

The PreSumm (Liu and Lapata, 2019b) model was used as a baseline in this study. We used the
### Table 7: The results of the next response prediction task.

<table>
<thead>
<tr>
<th>Model</th>
<th>R@1</th>
<th>R@2</th>
<th>R@5</th>
</tr>
</thead>
<tbody>
<tr>
<td>NRP</td>
<td>56.09</td>
<td>75.95</td>
<td>98.08</td>
</tr>
<tr>
<td>PRP</td>
<td>51.91</td>
<td>73.51</td>
<td>95.64</td>
</tr>
</tbody>
</table>

The model NRP refers to the task of predicting the next response given a dialog context, and the model PRP refers to the task of predicting the previous response given a dialog context.

PreSumm extractive summarization model which was pre-trained on the CNN/DM summarization dataset, and fine-tuned the model on the TWEETSUMM dataset. All the code and pre-trained models used in this study are publicly available.

The hyperparameters used for training the extractive summarization model are as follows:

- `ext_dropout=0.1`
- `lr=2e-3`
- `save_checkpoint_steps=5000`
- `batch_size=3000`
- `train_steps=50000`
- `accum_count=2`
- `warmup_steps=10000`
- `max_pos=512`

The checkpoint which produced the best performance on the validation dataset (checkpoint at step 35000) was used to initialize the PreSumm abstractive summarization model. The hyperparameters used for training the abstractive summarization model are as follows:

- `dec_dropout=0.2`
- `sep_optim=true`
- `lr_bert=0.002`
- `lr_dec=0.2`
- `save_checkpoint_steps=5000`
- `batch_size=140`
- `train_steps=100000`
- `accum_count=true`
- `use_bert_emb=true`
- `use_interval=true`
- `warmup_steps_bert=20000`
- `warmup_steps_dec=10000`
- `max_pos=512`
- `beam_size=5`

The checkpoint which produced the best performance on the validation dataset (checkpoint at step 55000) was used to generate summaries on the test dataset.

#### B.3 BART models

As a fully abstractive summarization algorithm, we used the BART model (Lewis et al., 2019) in this study. We use a lightweight variant of BART, named DistilBART provided by HuggingFace (Wolf et al., 2019) library. This instance of DistilBART is fine-tuned on the extreme summarization (XSum) task, and we fine-tune this model on the TWEETSUMM dataset. The code used for the fine-tuning is publicly available.

The hyperparameters used for training the DistilBART model are as follows:

- `train_batch_size=4`
- `eval_batch_size=4`
- `num_train_epochs=6`
- `model_name_or_path=sshleifer/distilbart-xsum-12-6`
- `learning_rate=3e-5`
- `val_check_interval=0.1`
- `max_source_length=512`
- `max_target_length=80`

#### C Sample summaries with corresponding QA questions

Figure 2 shows an example of a TWEETSUMM human-generated abstractive summary along with machine-generated summaries and their corresponding QA questions. Upon acceptance of the paper, TWEETSUMM release will include the set of questions that were generated as part of the human evaluation task in the Results section.

---

6https://github.com/nlpyang/PreSumm

7https://huggingface.co/sshleifer/distilbart-cnn-12-6

8https://github.com/huggingface/transformers/tree/master/examples/seq2seq
An awful smell in a flight

Sample QA Questions
Does the summary specify the customer is complaining about bad smell in his flight?
Does the summary specify the agent asked to contact in-flight crew member on duty for assistance?
Does the summary specify the customer asked to change seat in rebooking?
Does the summary specify the agent apologized for the discomfort?

Automated abstractive summary
BART
Customer is complaining about the smell on flight 1287 from Miami to LaGuardia. Agent requests to reach out to a flight attendant to address the odor in the aircraft.

Automated extractive summaries
NRP
Customer Flight1287 from Miami to LaGuardia smells awful. Every person getting on the flight is complaining. Agent Unfortunately, our First Class Cabin is full on our DL1287 flight for today, Allie. Please reach out to a flight attendant to address the odor in the aircraft.

LEAD
Customer Flight1287 from Miami to LaGuardia smells awful. It's really really bad. Agent Allie. I am very sorry about this. Please reach out to a flight attendant to address the odor in the aircraft.

CES
Customer Flight1287 from Miami to LaGuardia smells awful. They told us to rebook, then told us the original flight was still departing. Agent Unfortunately, our First Class Cabin is full on our DL1287 flight for today, Allie. You may seek further assistance by reaching out to one of our in-flight crew members on duty.

A Red Eye Removal issue

Sample QA Questions
Does the summary specify the customer asks to do red eye removal?
Does the summary specify the customer is using Lightroom CC?
Does the summary specify the agent sent an article containing the required information?
Does the summary specify the agent explained the released version contains all the features of the old version?
Does the summary specify the agent suggested the customer to report a complaint so the engineering team will get an alert and help?

Automated abstractive summary
BART
Customer is asking how to do red eye removal in Lightroom CC. Agent is looping their expert team to help answer the question.

Automated extractive summaries
NRP
Customer Can you tell me how to do Red Eye Removal in Lightroom CC? I just moved to it and don't see the Red Eye Removal tool. Agent Hi Bob, here is a link to show you to use the Red eye removal in Lightroom CC: [URL]. I am keeping our expert team to help answer your question.

LEAD
Customer Can you tell me how to do Red Eye Removal in Lightroom CC? I just moved to it and don't use the Red Eye Removal tool. Agent Hi Bob, here is a link to show you to use the Red eye removal in Lightroom CC: [URL]. Please let us know if you have any questions or need further help.

CES
Customer Can you tell me how to do Red Eye Removal in Lightroom CC? I wish a list of features missing in Lightroom CC would have been noted before I migrated my library. Agent Hi Bob, this feature is not available in Lightroom CC as of now, however you may suggest it as a feature here: [URL]. We have released Lightroom Classic CC which has all the features the old Lightroom CC 2015.12 had, you can check this article to see the differences between LR Classic and the new Lightroom CC: [URL].

Figure 2: Two ground-truth summaries with corresponding automated summaries and QA questions
PLEASE READ:

Overview

The following task is focused on customer care dialog summarization. You are asked to carefully read a dialog (taken from Yahoo) between a support agent implementing a repair and a customer. Then you are asked to highlight the most salient content expressed in the dialog both by the agent and by the customer. Finally you are asked to write one sentence summarizing the reasons the customer contacted the support center, and one sentence summarizing the agent's response.

Important instructions for the highlighting task

- You are asked to select between 2-3 sentences that describe the most important messages the customer conveyed. Focus on sentences representing a problem, complaint, or a request the customer expressed.
- You are asked to select between 2-3 sentences representing the agent response to the customer. Focus on actual solutions and not on apologies or gratitude expressions.
- Please between 5-8 sentences (no fewer no less) Focus on your selection only on informative content. Specifically on requests, problem descriptions and solutions. Not on apologies, gratitude, expressions of anger or frustration. Here are some examples of uninformative sentences that should not be selected:
  - Agent: “I’m sorry to hear that.”
  - Agent: “I’ll try to investigate this behavior.”
  - Customer: “Please send me a CM and we will get back to you.”
  - Agent: “Customer: fantastic! That is the worst customer support!”
  - Customer: “Help with MAP is the going back to the 2nd world country”
  - Customer: “Nokia 6800 company”
- After you complete to highlight sentences, please go over the selected ones and make sure it is possible to get the gist of the dialog without reading all the other ones in the dialog. If not, please amend your selection.

Important instructions for the short summaries: Write task generation

- You are asked to write two sentences:
  - The first sentence summarizes the customer interaction.
  - The second sentence summarizes the agent response.
- Make sure the text you are writing is informative. Please focus on requests, problem descriptions and solutions. Don’t summarize your personal opinion on the interaction.
- Here are some examples of uninformative short summaries:
  - “Customer talks about the 10 day return policy for headphones.”
  - “Explains headphones are excluded from returns and accept returns are only for damaged items.”
  - “Customer says the invoice is running at 100% CPU.”
  - “Agent stated that without completing data protection, they are not able to log this against their account and check for error.”

Figure 3: Annotation interface - Instructions for the summary generation task

Figure 4: Annotation interface - A dialog presented to annotators
Figure 5: Annotation interface - Annotators are asked to highlight salient sentences (for the extractive summary)