A Guide To Effectively Leveraging LLMs for Low-Resource Text Summarization: Data Augmentation and Semi-supervised Approaches

Gaurav Sahu[1,2] **Olga Vechtomova**[1] **Issam H. Laradji**[1,2,3]
[1]Cheriton School of Computer Science, University of Waterloo, Canada
[2]ServiceNow Research
[3]University of British Columbia, Canada
gaurav.sahu@uwaterloo.ca

Abstract

Existing approaches for low-resource text summarization primarily employ large language models (LLMs) like GPT-3 or GPT-4 at inference time to generate summaries directly; however, such approaches often suffer from inconsistent LLM outputs and are difficult to adapt to domain-specific data in low-resource scenarios. In this work, we propose two novel methods to effectively utilize LLMs for low-resource text summarization: **1)** *MixSumm*, an LLM-based data augmentation regime that synthesizes high-quality documents (short and long) for *few-shot text summarization*, and **2)** *PPSL*, a prompt-based pseudolabeling strategy for sample-efficient *semi-supervised text summarization*. Specifically, *MixSumm* leverages the open-source `LLaMA-3-70b-Instruct` model to generate new documents by mixing topical information derived from a small seed set, and *PPSL* leverages the `LLaMA-3-70b-Instruct` model to generate high-quality pseudo-labels in a semi-supervised learning setup. We evaluate our methods on the TweetSumm, WikiHow, and ArXiv/PubMed datasets and use L-Eval, a LLaMA-3-based evaluation metric, and ROUGE scores to measure the quality of generated summaries. Our experiments on extractive and abstractive summarization show that *MixSumm* and *PPSL* achieve competitive ROUGE scores as a fully supervised method with 5% of the labeled data. We release our codebase here: https://github.com/ServiceNow/text-summarization-with-llms/

1 Introduction

Text summarization is a crucial task in today's data-driven era, with applications ranging from news digests to summarizing scientific papers to summarizing customer chatlogs in enterprises (Cohan and Goharian, 2017; Zhong et al., 2020; Goyal et al., 2022; Feigenblat et al., 2021). Modern summarization systems can be broadly categorized into two types: abstractive, where the generated summaries are concise *paraphrases* of the source text (Barzilay and McKeown, 2005; Nallapati et al., 2016), and extractive, which select and arrange existing sentences in the source text (Wong et al., 2008; Kryściński et al., 2019). While abstractive methods produce more fluent and natural-sounding summaries–particularly beneficial for longer documents, extractive methods are valued for their simplicity and reliability in preserving factual accuracy; however, the performance of these summarization systems is often constrained by the availability and diversity of training data.

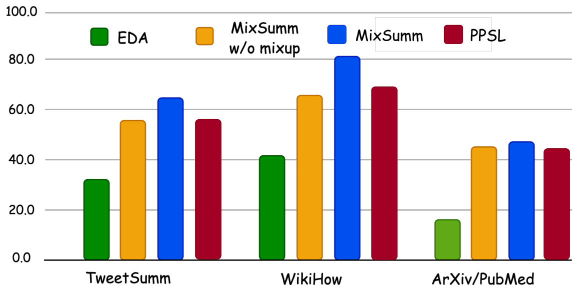
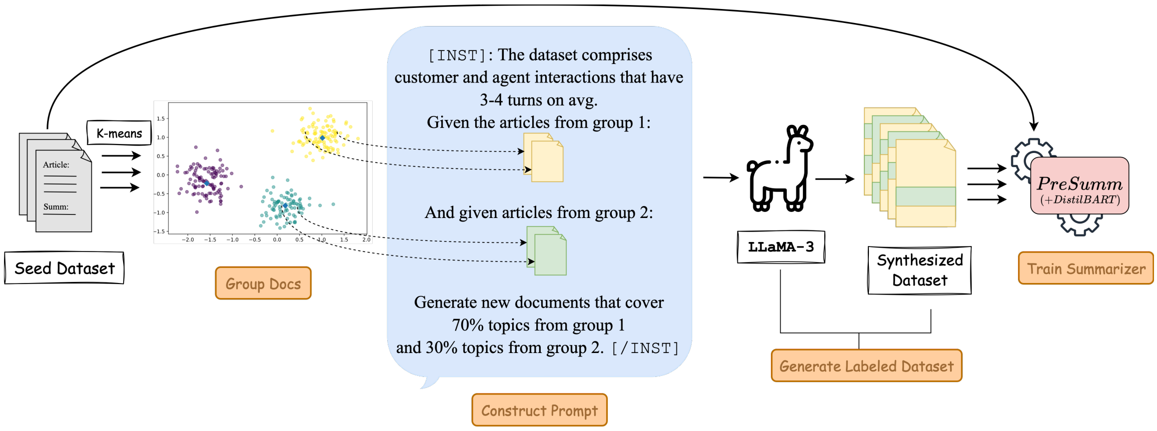


Figure 1: **L-Eval scores of different methods on low-resource extractive text summarization.** The proposed *MixSumm* approach generates new documents by combining topics from multiple examples and outperforms other methods, including a strong LLM-based DA method (*MixSumm* w/o mixup) and a prompt-based semi-supervised approach (PPSL).

Data augmentation (DA) has been successfully used to address data scarcity, mitigate data annotation costs, and enhance model robustness in various natural language processing (NLP) tasks like text classification, summarization, and grammatical error correction (Wei and Zou, 2019; Feng et al., 2021; Wang et al., 2022). Traditional augmentation methods involving synonym replacement, sentence shuffling, and back-translation are effective to some extent, but they quickly saturate as they do not fully capture the semantic nuances of the text; however, the recent surge in the development of LLMs like

Findings of the Association for Computational Linguistics: NAACL 2025, pages 1584–1603
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics



[INST]: The dataset comprises customer and agent interactions that have 3-4 turns on avg.
Given the articles from group 1:

And given articles from group 2:

Generate new documents that cover 70% topics from group 1 and 30% topics from group 2. [/INST]

Figure 2: ***MixSumm* pipeline.** We first group the documents into T groups using the k-means algorithm. Then, we construct the prompt for `LLaMA-3-70b-Instruct` by including documents from different groups and instructing the LLM to mix information from multiple topics when generating the new documents. Finally, we train a PreSumm extractive summarizer (Liu and Lapata, 2019) on the combined seed and the synthesized dataset. For abstractive summarization, we add a DistilBART model after PreSumm.

GPT-4 (Achiam et al., 2023), LLaMA-3 (Touvron et al., 2023), and Claude-3 (Anthropic, 2024), has given birth to the paradigm of LLM-based data augmentation techniques (Dai et al., 2023; Ding et al., 2024) that can generate contextually rich textual augmentations to enhance the performance of various NLP models such as dialog modeling systems (Chintagunta et al., 2021; Wan et al., 2022) and text classifiers (Yoo et al., 2021; Sahu et al., 2022). Real-life scenarios also often have a small labeled set alongside a large pool of unlabeled data, and semi-supervised learning (SSL) has been successfully used in such scenarios for images and text classification (Guillaumin et al., 2010; Gong et al., 2016; Liu et al., 2020; Miyato et al., 2016; Xu et al., 2017).

Despite the recent advancements in LLMs, neither LLM-based DA methods nor LLM-based semi-supervised methods have been extensively explored for low-resource text summarization. Therefore, in this work, we propose two novel methods to effectively utilize LLMs for low-resource text summarization: **1)** *MixSumm*, an LLM-based data augmentation technique for few-shot text summarization, and **2)** Prompt-based Pseudo-labeling for Semi-supervised Learning (*PPSL*), a pseudo-labeling strategy for sample-efficient semi-supervised text summarization. More specifically, *MixSumm* is a two-stage prompt-based data augmentation approach that **first** instructs an LLM to synthesize diverse documents by mixing topical information derived from a small set of seed documents, and **then** generates summaries for the synthesized documents. On the other hand, *PPSL* is a multi-step pseudo-labeling strategy for semi-supervised learning that generates high-quality pseudo-labels and selects most informative samples in an SSL pipeline.

To evaluate the effectiveness of our proposed framework, we conduct extensive experiments on the TweetSumm (Feigenblat et al., 2021), the WikiHow (Koupaee and Wang, 2018), and the ArXiv/PubMed (Cohan et al., 2018) text summarization datasets. We use the open-source `LLaMA-3-70b-Instruct` LLM for our tasks instead of a closed-source LLM like the GPT family of LLMs. For evaluation, we use the standard ROUGE scores (Lin, 2004) as well as L-Eval, an open-source version of the promising LLM-based evaluator for text summarization, G-Eval (Liu et al., 2023b). Our experiments demonstrate that *MixSumm* and *PPSL* outperform strong data augmentation and semi-supervised baselines for low-resource summarization setups *and* we show a knowledge distillation effect, where the knowledge of a LLaMA-3-70b model is distilled into the a much smaller summarization model using BERT$_{base}$ and DistilBART backend (with 110M and 306M parameters, respectively).

To summarize the contributions of our work: **1)** we propose *MixSumm*, a novel prompt-based data augmentation framework for the challenging low-resource setup of 50-shot text summarization, **2)** we propose *PPSL*, a novel pseudo-labeling strategy for sample-efficient semi-supervised text summarization, **3)** we show the effectiveness of

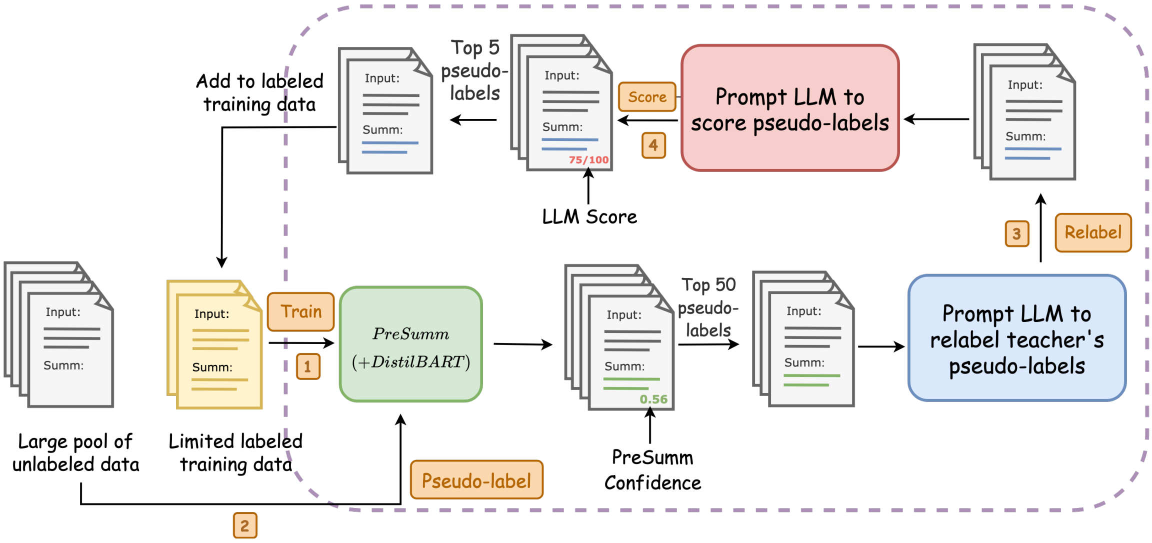


Figure 3: ***PPSL* pipeline.** **Step 1:** train a teacher model M on the limited labeled dataset. **Step 2:** generate pseudo-labels for the unlabeled set with M and shortlist 50 based on teacher confidence (see Equation 2). **Step 3:** prompt an LLM to summarize the shortlisted documents. **Step 4:** score the pseudo-labels in Stage 3 by prompting an LLM and select the top 5. These summaries are then added to the training data for the next cycle.

`LLaMA-3-70b-Instruct`, an open-source LLM, instead of using expensive closed-source LLMs like GPT-4, and **4)** we demonstrate effective knowledge distillation from LLaMA-3-70B (70B parameters) to BERT and DistilBART-based summarization models with 110M and 306M parameters.

2 Related Work

LLM-based Text Summarization. Fabbri et al. (2020) use round-trip back-translation to improve BART's abstractive summarization performance. On the other hand, Dou et al. (2021) propose GSum, a fully supervised transformer-based architecture that can use a guidance signal from an external source for improved abstractive text summarization. Goyal et al. (2022) employ zero-shot prompting on GPT-3 for open-ended news summarization and show that humans overwhelmingly prefer GPT-3 summaries over human summaries. Pu and Demberg (2023) use prompting on GPT-3 for controllable text summarization and show that while GPT-3 can follow simple constraints in the prompt like length, it shows a noticeably lower degree of change in styles compared to human-written summaries. Liu et al. (2024a) and Zhang et al. (2024) benchmark the zero-shot performance of LLMs on instruction-controlled summarization and news summarization. Chintagunta et al. (2021) use GPT-3 as a data annotator for 210-shot medical dialog summarization and show significant gains equivalent to using 6400 human-written labels. More recently, Liu et al. (2024b) fine-tune BART on LLM-generated summaries instead of human-generated summaries and show that LLMs are excellent references. Notably, these works prompt GPT-3 directly for summarization in their experiments. Except for the last two works, none of them use LLMs as data generators in low-resource setups. Additionally, they all use a closed-source LLM in their experiments. Zhang et al. (2023) propose an extract-then-generate method where they use in-context learning to generate extractive-summary-guided abstractive summaries. However, since they operate in a fully-supervised setting, the method suffers from scalability issues for large datasets. Mishra et al. (2023) propose LLM pseudo-labeling for semi-supervised dialog summarization, but our proposed *PPSL* method is more sample-efficient as we use fewer labeled and unlabeled examples.

LLM-based Distillation and Data Augmentation in NLP. A large body of recent work uses LLMs as data generators for distilling a large teacher model's knowledge into smaller models for training instruction-tuned models and chain-of-thought reasoning, while reducing human annotation load (Ho et al., 2023; Shum et al., 2023; Meng et al., 2023; Liu et al., 2023a; Peng et al., 2023). Bonifacio et al. (2022) use few-shot prompting to construct training datasets with query-document pairs for information retrieval. In the landscape of few-shot text classification, Yoo et al. (2021) propose GPT3Mix and Sahu et al. (2022, 2023) propose

PromptMix, where both the methods use LLMs as data generators and data labelers. We are inspired by the success of LLM-based DA for these diverse NLP tasks and adopt the best prompting practices based on these works. For instance, we generate diverse examples by mixing examples from different classes or groups as in GPT3Mix and PromptMix; and specify concrete criteria when using LLMs for generation and evaluation as in Pu and Demberg (2023) and Liu et al. (2024b). Furthermore, we conduct extensive experiments to test the capabilities of an open-source LLM, LLaMA-3-70b, for low-resource text summarization, instead of using closed-source LLM like GPT-3 and GPT-4. Finally, we also test our LLM-based DA strategy on extremely long documents.

3 Notations

We denote an annotated, many-shot summarization dataset as $\mathcal{D} = (d_i, s_i) \; \forall \; i \in \{1, \ldots, N\}$, where (d_i, s_i) denotes the i-th datapoint with input text document d_i and its ground truth summary s_i. We refer to the training, validation, and testing parts of the dataset as $\mathcal{D}_{train}$, $\mathcal{D}_{val}$, and $\mathcal{D}_{test}$, respectively. Given the many-shot training set $\mathcal{D}_{train}$, we construct a few-shot version of the dataset with k examples $\mathcal{D}_{F,train}$ and the unlabeled set $\mathcal{D}_{U,train}$ as follows:

Step 1. Given $\mathcal{D}_{F,train}$, we group the training articles by topics. We do not define the topics explicitly and identify T groups by applying the k-means algorithm on the document embeddings (where $k = T$)[1]. We use the SBERT encodings (Reimers and Gurevych, 2019) of the input documents as document embeddings[2]. If an input document exceeds SBERT's context window length of 512 tokens (roughly 300-400 English words), we chunk the document into smaller pieces and then average the chunk embeddings to obtain the final document embedding.

Step 2. We construct our k-shot dataset $\mathcal{D}_{F,train}$ by randomly sampling an equal number of datapoints from each of the T clusters so that $\mathcal{D}_{F,train}$ has k examples in total. In Section 6, we empirically show that our principled approach for constructing few-shot datasets is better than randomly sampling k examples from $\mathcal{D}_{train}$ as it provides better topical coverage.

Step 3. We randomly select m documents from $\mathcal{D}_{train} \setminus \mathcal{D}_{F,train}$ (without labels) to construct the unlabeled set of documents $\mathcal{D}_{U,train}$ to be used in the semi-supervised setup.

Problem Formulation (few-shot setup). Given a text summarization dataset $\mathcal{D}$: **1)** perform data augmentation on $\mathcal{D}_{F,train}$ to synthesize a labeled dataset $\mathcal{D}_{A,train}$, and **2)** train a text summarization model on the combined dataset $\mathcal{D}_{F+A,train}$.

Problem Formulation (semi-supervised setup). Given a text summarization dataset $\mathcal{D}$: **1)** perform SSL on $\mathcal{D}_{F,train}$ and $\mathcal{D}_{U,train}$ to obtain a pseudo-labeled dataset $\mathcal{D}_{F+U,train}$, and **2)** train a text summarization model on the combined dataset $\mathcal{D}_{F+U,train}$.

4 Methodology

4.1 *MixSumm* for Few-Shot Text Summarization

We now describe, *MixSumm*, a two-step approach for synthesizing labeled summarization documents. **First**, instruct an LLM to generate documents that cover multiple topics derived from a small set set. **Next**, we instruct an LLM to generate summaries for those documents. The following sections describe our two-step procedure in detail.

Step 1: Synthesizing New Documents. *First,* for every dataset, we manually write a short description that describes the type and approximate size of articles in the dataset. These descriptions enable our approach to be used in even zero-shot settings. *Next,* we construct T pairs of clusters $(c_i, c_j) \; \forall \; i, j \in 1, \ldots, T, i \neq j$, such that c_j is the most distant cluster from c_i. We use the centroids of the clusters obtained during k-means clustering in Section 3 for our computation. We also ensure that all cluster pairs are *unique* as $(c_i, c_j) \equiv (c_j, c_i)$. *Finally,* we combine the dataset description with k examples from each cluster and instruct the LLM to generate new examples that cover topics from both clusters. Specifically, we instruct the LLM to generate examples that contain $\alpha\%$ topics from the first cluster c_i and $(1 - \alpha)\%$ topics from the second cluster c_j, where α is sampled from a uniform distribution between 1-100. This is similar to applying the mixup algorithm (Zhang et al., 2018) in a natural language space and has proven highly effective for data augmentation in low-resource text classification setups (Yoo et al., 2021; Sahu et al.,

[1] to clarify, k in k-means is different from k in k-shot

[2] `sentence-transformers/all-mpnet-base-v2` model was used from the sentence-transformers library

2023). Prompt 1 in Appendix E shows the complete template for this step.

Step 2: Generating Summaries for the Synthesized Documents. Next, we instruct the LLM to generate extractive and abstractive summaries for the synthesized documents. **For extractive summaries,** we provide a generated document to the LLM and then instruct it to output a probability score for each sentence indicating whether that sentence should be included in the summary or not. We then rank the lines by the scores and choose the top-p lines, where p is the summary size and depends on the dataset. We truncate the input document if it exceeds the LLM's context window length. This approach ensures the extractiveness of the generated summary labels as it mimics PreSumm (Liu and Lapata, 2019), a strong baseline for extractive text summarization. **For abstractive summaries,** instead of passing the entire source document and prompting the LLM to generate a summary, we ask it to summarize the previously generated extractive summaries. This approach is faster than passing the source document and summarizing as our input context is significantly smaller. More importantly, it enhances the factual correctness of the summaries.

4.2 *PPSL* for Semi-Supervised Text Summariation

This section describes our approach for semi-supervised text summarization. As shown in Figure 2, we employ a teacher-student training framework and divide our pipeline into four steps, where we first train a teacher model on $\mathcal{D}_{F,train}$, then use it to generate pseudo-labels for $\mathcal{D}_{U,train}$, prompt the LLM to relabel the teacher's pseudo-labels from the previous step, and lastly score the new pseudo-labels with an LLM and select top 5 to include in the next training cycle.

Step 1: Training the Teacher Model First, we train a fully-supervised model M (teacher) on the set of available labeled examples $\mathcal{D}_{F,train}$. We use PreSumm (Liu and Lapata, 2019) as our extractive summarizer as it has been shown to perform well for extractive summarization. Notably, PreSumm reformulates the task of generating extractive summaries to binary classification, where, for each sentence in the input document, the model predicts if it will be present in the output summary. Then, the model combines the top-n sentences with the highest probabilities in their order of appearance in the original text to construct the extractive summary. **For abstractive summarization**, we follow an *extractive-then-abstractive* approach and add a DistilBART model that summarizes PreSumm's summary. The rest of the subsequent steps remain unchanged.

Step 2: Generating Pseudo-labels using the Teacher Model We use the teacher model M to generate pseudo-labels for the unlabeled set $\mathcal{D}_{U,train}$. Next, we shortlist a subset of 50 pseudo-labels with the highest teacher confidence [3]. We describe confidence computation in detail in Appendix B We will show in Section 6 that shortlisting a subset of pseudo-labels helps make our method more sample-efficient, as we avoid relabeling a large unlabeled pool. This ultimately minimizes our LLM usage cost in the subsequent steps.

Step 3: LLM Relabeling of Teacher's Pseudo-labels After selecting the top 50 pseudo-labels using teacher confidence defined in Equation 2, we prompt the LLM to generate a summary for each shortlisted unlabeled example. This effectively relabels the pseudo-label from Step 2. Specifically, we follow the prompt template in Figure 6a when generating summaries, which uses the same mechanism as the teacher M, i.e., for extractive summarization, we instruct the LLM to output probabilities for each sentence in the input document and then concatenate the top-n lines in their order of appearance in the input text and for abstractive, we further ask the LLM to summarize the extractive summary.

Step 4: LLM Scoring of Pseudo-labels In the last step of our *PPSL*, we prompt LLaMA-3 as shown in Figure 6b to output a rating between 0-100 for the pseudo-labels from Step 3. Finally, we choose the top 5 pseudo-labeled examples with the highest LLM scores and add them to the existing labeled set. We repeat Steps 1-4 N_{cycles} times to improve the initial summarization model M and use the model obtained after the last cycle for generating summaries for the unseen test set.

5 Experimental Setup

5.1 Datasets

We use three popular datasets in this work for extractive text summarization.

[3] We experiment 5, 10, 20, 50, and 75 and find 50 to be optimal.

	TweetSumm	WikiHow	ArXiv/PubMed
# Train	858	168,000	300,000
# Valid	100	6,000	12,500
# Test	100	6,000	12,500
Avg. Doc. Length	245.01	579.8	4203.4

Table 1: Statistics of the text summarization datasets used in our experiments. **Note:** Avg. doc. length is reported in the number of tokens.

1) TweetSumm (Feigenblat et al., 2021) is a real-world customer service dataset that has 1100 conversations between a customer and an agent, and each conversation has three human-annotated extractive summaries. The training set has 858 dialogs, and the validation and test sets have 100 examples each. **2) WikiHow** (Koupaee and Wang, 2018) contains WikiHow articles with their headlines as abstractive summaries. The dataset has over 180k articles, with around 168k training articles and 6000 test and validation articles. **3) ArXiv/PubMed** (Cohan et al., 2018) is a collection of scientific articles from PubMed and ArXiv with their abstracts as summaries. The dataset has ~325k articles, with nearly 300k training articles and 12.5k test and validation articles.

Table 1 summarizes the dataset statistics. Since WikiHow and ArXiv/PubMed datasets do not have extractive labels, we follow the same steps as the original PreSumm paper (Liu and Lapata, 2019) and construct an extractive summary that maximizes the ROUGE score between the obtained extractive summary and the ground-truth abstractive summary. We chose the three datasets above as they cover diverse scenarios, from relatively short real-world customer-agent conversations in the TweetSumm dataset to long scientific articles in the ArXiv/PubMed dataset. We report the training implementation details in Appendix C.

5.2 Evaluation.

We evaluate the summary quality of the models using the following metrics:

ROUGE Scores. We use ROUGE-1 (R-1), ROUGE-2 (R-2), and ROUGE-L (R-L) F1 scores (Lin, 2004) for evaluation, where R-1 and R-2 measure the unigram and bigram overlap between the predicted and the ground truth summaries, respectively, while R-L also considers the order of n-grams. We use the `pyrouge` Python package to compute ROUGE scores in our setup and report them in Table 2.

L-Eval Scores. In addition to ROUGE, we use an LLM-based evaluation metric for our task. Specifically, we use LLaMA-Eval (L-Eval), an open-source variant of the G-Eval metric (Liu et al., 2023b), where we prompt `LLaMA-3-70b-Instruct` instead of a GPT model. We use L-Eval as it better aligns with human preferences for text summarization, compared to ROUGE scores and other model-based evaluation metrics, such as BERTScore and BARTScore (Zhang et al., 2019; Yuan et al., 2021). It is also not biased towards LLM-generated content; however, since LLM-inference speed is low for long documents, we did not compute L-Eval scores during training and only computed them during final testing. When computing L-Eval scores, we provide the LLM with the input article and a (generated) summary and instruct it to score the summary on a scale of 1-10 (see Prompt 2 in Appendix E for the full L-Eval prompt template). Formally, given a test article A and a summary s, we compute the L-Eval score as follows:

$$\text{L-Eval}(A, s) = \sum_{r=1}^{10} p_r \cdot r, \qquad (1)$$

where p_r is the probability of generating the rating r. In practice, we can only look at the probabilities of top-5 tokens for `LLaMA-3-70b-Instruct`, so we assign a probability of 0 to the remaining ratings (that did not appear in the top-5).

In total, computing *test* L-Eval scores for all the summarization models included in Table 2 took ~5.6 hrs for TweetSumm, ~2.1 days for WikiHow, and ~6 days for the ArXiv/PubMed dataset.

5.3 Baselines

We run the following baselines: **1) *MixSumm* (Ours).** We augment $\mathcal{D}_{F,train}$ using the proposed *MixSumm* approach then train a summarization model on $\mathcal{D}_{F+A,train}$. We also run two variants of this baseline to determine the effect of applying data augmentation and mixup, denoted by ***MixSumm* w/o Aug.** and ***MixSumm* w/o Mixup** respectively. **2) Easy Data Augmentation (EDA).** We use an edit-based data augmentation technique (Wei and Zou, 2019) to construct $\mathcal{D}_{A,train}$ instead of using *MixSumm*. Specifically, we apply the EDA technique to each sentence in an article to construct a new example. **3) *MixSumm* (rand.).** Same as **1)** but $\mathcal{D}_{F,train}$ is constructed by randomly selecting k examples from the full

	TweetSumm				WikiHow				ArXiv/Pubmed			
Method	**R-1 (%)**	**R-2 (%)**	**R-L (%)**	**L-Eval (%)**	**R-1 (%)**	**R-2 (%)**	**R-L (%)**	**L-Eval (%)**	**R-1 (%)**	**R-2 (%)**	**R-L (%)**	**L-Eval (%)**
Extractive Summarization												
Oracle	*65.7*±0.3	*56.6*±0.4	*64.9*±0.4	*86.2*±0.3	*30.5*±0.5	*8.7*±0.3	*19.2*±0.6	*87.3*±0.1	*34.6*±0.4	*12.4*±0.2	*19.6*±0.4	*78.1*±0.5
TSL (50:500)	49.0	37.7	48.2	-	-	-	-	-	-	-	-	-
TSL (500:500)	59.0	48.3	58.2	-	-	-	-	-	-	-	-	-
EDA	51.1±0.7	39.2±0.9	53.0±0.2	34.3±1.2	23.4±0.5	4.1±0.3	13.0±0.5	42.1±0.8	26.2±1.1	7.9±1.0	13.1±0.6	17.2±0.5
PPSL (50:250)	58.4±1.2	50.1±0.3	59.1±1.2	56.3±0.9	26.0±0.2	6.9±0.3	15.1±0.2	69.3±2.1	29.0±0.5	9.4±0.7	17.4±0.3	49.3±1.4
MixSumm (rand.)	58.6±3.2	50.6±2.1	59.7±2.3	60.3±0.9	26.4±1.0	7.5±1.2	15.8±0.2	72.5±1.2	30.7±1.7	**10.6**±1.5	**18.5**±1.1	48.4±1.1
w/o Aug.	49.4±0.7	36.9±1.0	49.0±0.2	31.5±0.5	21.3±0.4	3.2±0.4	11.4±0.5	34.2±1.5	23.4±1.1	7.5±1.4	12.3±0.8	13.5±1.2
MixSumm (ours)	**59.1**±1.7	**52.7**±1.6	**60.5**±1.3	**65.3**±1.2	**27.3**±2.1	**7.8**±1.3	**16.6**±1.8	**81.1**±1.7	**31.2**±1.2	**10.7**±1.1	**18.3**±1.1	**53.1**±0.5
w/o Mixup	56.1±1.1	47.3±1.2	55.3±1.1	57.3±0.5	25.7±1.4	6.2±1.2	14.7±0.7	67.3±2.1	28.4±1.9	8.3±1.3	16.8±1.6	52.3±1.2
w/o Aug.	50.1±0.6	38.1±1.0	49.9±0.6	32.3±3.1	21.9±0.3	3.5±0.2	12.1±0.9	33.3±1.7	24.1±0.9	7.9±1.0	12.7±0.5	19.0±2.5
LLaMA-3 (0-shot)	50.3±0.5	47.7±0.4	49.9±0.3	52.3±1.2	12.2±0.2	2.7±0.5	8.1±0.4	32.3±0.3	23.6±0.2	4.6±0.7	15.4±0.3	**38.4**±0.5
LLaMA-3 (1-shot)	51.7±0.2	49.2±0.3	51.9±0.3	58.7±1.1	14.3±0.2	4.1±0.5	10.6±0.2	39.4±0.5	**32.6**±0.4	**6.5**±0.7	**17.2**±0.3	38.3±1.8
LLaMA-3 (5-shot)	**62.4**±0.5	**54.3**±0.7	**60.3**±1.1	**67.5**±0.6	**28.7**±0.3	**7.5**±0.9	**17.1**±0.3	**71.3**±0.4	-	-	-	-
Abstractive Summarization												
Oracle	*44.7*±0.2	*20.1*±0.4	*36.8*±0.2	*72.3*±0.6	*28.7*±0.3	*6.2*±0.7	*13.6*±0.4	*78.4*±0.8	*28.4*±0.2	*10.2*±0.4	*15.8*±0.8	*64.3*±0.5
EDA	41.5±1.2	15.0±0.8	32.2±1.1	44.2±1.6	14.7±1.8	3.2±1.0	6.8±1.5	40.5±1.4	16.3±1.5	5.9±0.8	8.1±1.7	36.8±1.3
PPSL (50:250)	42.7±1.5	18.1±1.1	33.8±1.3	58.1±1.3	**26.9**±1.8	**5.7**±1.0	**12.1**±1.5	62.1±1.4	26.7±1.5	9.5±0.8	**13.8**±1.7	61.3±1.3
MixSumm (ours)	**43.1**±1.1	**18.4**±1.5	**34.7**±1.0	**62.3**±1.4	26.7±1.7	5.3±0.9	11.3±1.4	**67.5**±1.3	**27.1**±1.4	**9.8**±0.7	13.5±1.6	**61.4**±1.2
w/o Mixup	37.5±1.0	16.0±1.3	31.2±0.9	58.2±1.2	23.2±1.5	4.6±0.8	9.8±1.2	58.7±1.1	23.6±1.2	8.5±0.6	11.7±1.4	55.8±1.0
w/o Aug.	23.7±1.2	10.1±1.7	18.3±1.1	34.9±1.5	14.0±1.9	2.9±1.0	6.2±1.6	36.4±1.4	14.5±1.3	5.4±0.8	7.4±1.8	19.2±1.3
LLaMA-3 (0-shot)	37.5±1.1	13.4±0.7	21.3±0.3	42.0±1.2	11.3±0.4	2.5±0.2	7.6±1.1	34.7±0.2	20.4±1.2	2.3±0.7	9.6±1.3	26.7±1.5
LLaMA-3 (1-shot)	37.8±1.0	13.5±0.8	21.5±0.4	41.7±1.3	11.5±0.5	2.4±0.2	7.8±1.0	34.9±0.3	20.2±1.1	2.4±0.6	9.5±1.2	**26.9**±1.4
LLaMA-3 (5-shot)	**44.2**±0.9	**19.8**±1.1	**36.1**±1.2	**64.4**±0.7	**26.2**±0.6	**5.6**±0.4	**12.1**±0.6	**69.3**±1.2	-	-	-	-

Table 2: **Summarization Results.** Comparison of different text summarization models on TweetSumm, WikiHow, and ArXiv/PubMed datasets. We report ROUGE-1 (R-1), ROUGE-2 (R-2), ROUGE-L (R-L) F_1 scores, and L-Eval scores. We report the mean±std. performance across 5 different seeds. Refer to Appendix C and Section 5.3 for metric and implementation details. **Note.** TSL results are reported from Zhuang et al. (2023). For EDA and *MixSumm* we use a 50-shot $\mathcal{D}_{F,train}$ and generate 1000 examples as $\mathcal{D}_{A,train}$. **Bold** denotes the best-performing model in a given block and highlight denotes the *overall* best-performing model. For the ArXiv/PubMed dataset, we could fit only 2 documents into LLaMA-3's context (1 from $\mathcal{D}_{F,train}$ + 1 generated), so we do not report LLaMA-3 (5-shot).

training set instead of selecting examples from the T clusters. We also run ***MixSumm* (rand.) w/o Aug.** where we do not perform any data augmentation. **4) Teacher Student Learning (TSL).** A semi-supervised setup proposed by Zhuang et al. (2023) that employs a teacher-student learning framework similar to us except they do not use LLM-based pseudo-labeling or relabeling. We report the performance of the TSL (50:500) and TSL (500:500) models [4]. **5) *PPSL*.** Proposed semi-supervised approach using teacher confidence and prompt-based pseudolabel scoring for text summarization. We report results for the PPSL (50:250) setting that uses `LLaMA-3-70b-Instruct`. **6) LLaMA-3-70b (k-shot.)** An in-context learning-based approach where we prompt `LLaMA-3-70b-Instruct` with k examples randomly selected from $\mathcal{D}_{F,train}$ and then instruct it to summarize a test article. We use the same prompt as the one we use for summarizing articles (Prompt 3 in Appendix E), except we remove the group information and directly populate it with k examples. **7) Oracle.** A fully supervised model trained on the complete training set $\mathcal{D}_{train}$ to gauge the upper-bound performance for this task.

6 Results

6.1 *MixSumm* Generates Diverse Documents.

Table 7 shows qualitative examples generated by EDA, *MixSumm* w/o mixup and *MixSumm* in Table 7. In the context of Table 7, we note that w/o mixup, *MixSumm* generates decent quality documents, but it only covers a single topic (phone/-electronic device-related sentences.) *MixSumm*, on the other hand, generated an example that contains mention of terms from two topics (flight as well as a device-related issue.) EDA generates the lowest-quality documents with grammatical errors and other artifacts. However, we note that regardless of the quality of the original document, LLaMA-3-70b generates a high-quality summary in all cases.

Comparison w/ Other DA methods. From Table 2, we note that *MixSumm* achieves significantly higher L-Eval and ROUGE scores for both extractive and abstractive summarization tasks. This demonstrates the superior generation ability of LLMs compared to a simple edit-based DA technique like EDA. Next, we compare *MixSumm* with *MixSumm* w/o Mixup, a strong LLM-based data augmentation baseline, and note that removing the mixup component from *MixSumm* significantly

[4] TSL (m:n) denotes access to m labeled examples and n unlabeled examples

lowers ROUGE and L-Eval scores across the board (as verified by a T-test).

6.2 Effect of Clustering Documents.

We perform a student's T-Test comparing results from *MixSumm* and *MixSumm* (rand.) and note that while ROUGE scores for *MixSumm* are generally higher than *MixSumm* (rand.), the differences are *not* significant. The only exception was R-2 scores on TweetSumm, where *MixSumm* outperforms *MixSumm* (rand.) by 2.1 points (R-2 of 52.7 v/s 50.6). On the other hand, the difference in L-Eval scores for the two methods was found to be significant by the T-test for all the datasets. This further suggests that ROUGE scores might not be able to capture the semantic correctness of the generated summaries and highlights the importance of an LLM-based evaluator that can discern between nuanced semantics in natural language text. We observe a similar trend after removing the augmentation component from both methods (*MixSumm* w/o Aug. v/s *MixSumm* (rand.) w/o Aug.).

Overall, we conclude that *MixSumm* is better than *MixSumm* (rand.), and we should include diverse examples, if possible, in the prompt as it leads to direct improvements in generation quality.

6.3 DA v/s SSL Methods

Comparing *MixSumm* with *PPSL* and TSL in Table 2, we note that our 50-shot *MixSumm* and *MixSumm* (rand.) methods outperform TSL (50:500), which uses 50 labeled examples and 500 unlabeled examples. Next, our two methods outperform TSL (500:500) on all the metrics except the R-1 score (where the different was not found to be significant). Overall, *MixSumm* is better than TSL for extractive summarization in extreme data-scarce settings. Next, we note that *MixSumm* achieves slightly higher ROUGE scores and significantly higher L-Eval scores than PPSL (50:250) for extractive summarization; however, for abstractive summarization, *MixSumm* and *PPSL* achieve very similar performance for the three datasets. Overall, we conclude that prompt-based data augmentation might be better than using a semi-supervised method for extractive summarization in data-scarce setups, but both methods are equally performant for abstractive summarization.

6.4 Knowledge Distillation from LLaMA-3

First, we note that increasing the number of examples for the LLaMA-3 method leads to expected improvements in performance except L-Eval scores on the ArXiv/PubMed dataset, where 0-shot and 1-shot LLaMA-3 models achieve similar L-Eval scores. This may suggest that LLaMA-3 struggles with understanding very long documents. Next, we note that 0-shot LLaMA-3 outperforms 50-shot *MixSumm* w/o Aug baseline on the TweetSumm dataset in terms of ROUGE scores and L-Eval scores, and it achieves competitive results on ArXiv/PubMed. Lastly, we note that *MixSumm* achieves competitive performance against LLaMA-3 as a summarizer for both extractive and abstractive tasks, whereas, *PPSL* is competitive with LLaMA-3 on only the abstractive task. Additionally, we note that our methods achieve comparable ROUGE scores to the Oracle model despite using just 50 labels compared to 1000 examples used by the oracle (95% less). Overall, we conclude that both *MixSumm* and *PPSL* are highly performant models compared to LLaMA-3-70b model, demonstrating effective distillation effect from LLaMA-3-70b to BERT- and DistilBART-based models. We include additional ablation studies in Appendix D that demonstrate the sample efficiency of *PPSL* and show the importance of relabeling and the specific pseudo-labeling strategy used in *PPSL*.

7 Conclusion

In this work, we focus on low-resource text summarization and propose two novel approaches to effectively employ an LLM for the task: *MixSumm*, a two-step data augmentation method for few-shot summarization, and *PPSL*, a multi-step prompt-based semi-supervised framework for sample-efficient semi-supervised text summarization. Our experiments show that our methods are better than existing approaches for low-resource summarization and that they knowledge transfer from a large teacher model `LLaMA-3-70b-Instruct` into much smaller BERT- and DistilBART-based models. LLM-based approaches are underexplored for low-resource text summarization, and through this work, we hope to spark an interest in the research community to address various challenges of this task.

8 Limitations

We use `LLaMA-3-70b-Instruct` for our experiments, which has a context window size of 8192 tokens, so it is not possible to fit many long documents in the model's context (like articles in the

ArXiv/PubMed dataset). We can explore using position interpolation (PI) to increase the context window length of LLaMA (Chen et al., 2023) or switch to more recent LLaMA-3.1 family of models.

Currently, we only consider text summarization for the English language. Moving forward, we can expand our method to multiple languages. More research on efficiently handling long documents during the training process is also needed, as we currently rely on a chunk-and-summarize subroutine to train our models for long documents, which results in significant delays in document processing. We can consider using alternative transformer architectures such as LongFormer (Beltagy et al., 2020) as PreSumm's backend.

9 Ethics Statement

We generate large textual datasets using LLMs, and even though we use an instruction-tuned model, we need to be careful about any bias it might exhibit, or any potentially harmful content that it might generate. Language model debiasing is a common potential solution to address this issue (Meade et al., 2021; Guo et al., 2022). Additionally, we suggest involving a human moderator if these systems are to be made public-facing.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

AI Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 1.

Regina Barzilay and Kathleen R McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Luiz Bonifacio, Hugo Abonizio, Marzieh Fadaee, and Rodrigo Nogueira. 2022. Inpars: Data augmentation for information retrieval using large language models. *arXiv preprint arXiv:2202.05144*.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*.

Bharath Chintagunta, Namit Katariya, Xavier Amatriain, and Anitha Kannan. 2021. Medically aware GPT-3 as a data generator for medical dialogue summarization. In *Proceedings of the Second Workshop on Natural Language Processing for Medical Conversations*, pages 66–76, Online. Association for Computational Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Arman Cohan and Nazli Goharian. 2017. Scientific article summarization using citation-context and article's discourse structure. *arXiv preprint arXiv:1704.06619*.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, et al. 2023. Auggpt: Leveraging chatgpt for text data augmentation. *arXiv preprint arXiv:2302.13007*.

Bosheng Ding, Chengwei Qin, Ruochen Zhao, Tianze Luo, Xinze Li, Guizhen Chen, Wenhan Xia, Junjie Hu, Anh Tuan Luu, and Shafiq Joty. 2024. Data augmentation using llms: Data perspectives, learning paradigms and challenges. *arXiv preprint arXiv:2403.02990*.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. Gsum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842.

Alexander R Fabbri, Simeng Han, Haoyuan Li, Haoran Li, Marjan Ghazvininejad, Shafiq Joty, Dragomir Radev, and Yashar Mehdad. 2020. Improving zero and few-shot abstractive summarization with intermediate fine-tuning and data augmentation. *arXiv preprint arXiv:2010.12836*.

Guy Feigenblat, Chulaka Gunasekara, Benjamin Sznajder, Sachindra Joshi, David Konopnicki, and Ranit Aharonov. 2021. TWEETSUMM - a dialog summarization dataset for customer service. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 245–260, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*.

Chen Gong, Dacheng Tao, Stephen J Maybank, Wei Liu, Guoliang Kang, and Jie Yang. 2016. Multi-modal curriculum learning for semi-supervised image classification. *IEEE Transactions on Image Processing*, 25(7):3249–3260.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2022. News summarization and evaluation in the era of gpt-3. *arXiv preprint arXiv:2209.12356*.

Matthieu Guillaumin, Jakob Verbeek, and Cordelia Schmid. 2010. Multimodal semi-supervised learning for image classification. In *2010 IEEE Computer society conference on computer vision and pattern recognition*, pages 902–909. IEEE.

Yue Guo, Yi Yang, and Ahmed Abbasi. 2022. Autodebias: Debiasing masked language models with automated biased prompts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1012–1023.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14852–14882, Toronto, Canada. Association for Computational Linguistics.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Wojciech Kryściński, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Evaluating the factual consistency of abstractive text summarization. *arXiv preprint arXiv:1910.12840*.

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

Hanmeng Liu, Zhiyang Teng, Leyang Cui, Chaoli Zhang, Qiji Zhou, and Yue Zhang. 2023a. LogiCoT: Logical chain-of-thought instruction tuning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2908–2921, Singapore. Association for Computational Linguistics.

Quande Liu, Lequan Yu, Luyang Luo, Qi Dou, and Pheng Ann Heng. 2020. Semi-supervised medical image classification with relation-driven self-ensembling model. *IEEE transactions on medical imaging*, 39(11):3429–3440.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023b. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Yixin Liu, Alexander Fabbri, Jiawen Chen, Yilun Zhao, Simeng Han, Shafiq Joty, Pengfei Liu, Dragomir Radev, Chien-Sheng Wu, and Arman Cohan. 2024a. Benchmarking generation and evaluation capabilities of large language models for instruction controllable summarization. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4481–4501, Mexico City, Mexico. Association for Computational Linguistics.

Yixin Liu, Kejian Shi, Katherine He, Longtian Ye, Alexander Fabbri, Pengfei Liu, Dragomir Radev, and Arman Cohan. 2024b. On learning to summarize with large language models as references. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8647–8664, Mexico City, Mexico. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Nicholas Meade, Elinor Poole-Dayan, and Siva Reddy. 2021. An empirical survey of the effectiveness of debiasing techniques for pre-trained language models. *arXiv preprint arXiv:2110.08527*.

Yu Meng, Martin Michalski, Jiaxin Huang, Yu Zhang, Tarek Abdelzaher, and Jiawei Han. 2023. Tuning language models as training data generators for augmentation-enhanced few-shot learning. In *International Conference on Machine Learning*, pages 24457–24477. PMLR.

Nishant Mishra, Gaurav Sahu, Iacer Calixto, Ameen Abu-Hanna, and Issam H Laradji. 2023. Llm aided semi-supervision for extractive dialog summarization. *arXiv preprint arXiv:2311.11462*.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, 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*, pages 280–290.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Dongqi Pu and Vera Demberg. 2023. Chatgpt vs human-authored text: Insights into controllable text summarization and sentence style transfer. *arXiv preprint arXiv:2306.07799*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Gaurav Sahu, Pau Rodriguez, Issam Laradji, Parmida Atighehchian, David Vazquez, and Dzmitry Bahdanau. 2022. Data augmentation for intent classification with off-the-shelf large language models. In *Proceedings of the 4th Workshop on NLP for Conversational AI*, pages 47–57, Dublin, Ireland. Association for Computational Linguistics.

Gaurav Sahu, Olga Vechtomova, Dzmitry Bahdanau, and Issam H Laradji. 2023. Promptmix: A class boundary augmentation method for large language model distillation. *arXiv preprint arXiv:2310.14192*.

Kashun Shum, Shizhe Diao, and Tong Zhang. 2023. Automatic prompt augmentation and selection with chain-of-thought from labeled data. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12113–12139, Singapore. Association for Computational Linguistics.

Leslie N Smith. 2015. Cyclical learning rates for training neural networks. arxiv. *Preprint at https://arxiv.org/abs/1506.01186*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Dazhen Wan, Zheng Zhang, Qi Zhu, Lizi Liao, and Minlie Huang. 2022. A unified dialogue user simulator for few-shot data augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3788–3799, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yiming Wang, Qianren Mao, Junnan Liu, Weifeng Jiang, Hongdong Zhu, and Jianxin Li. 2022. Noise-injected consistency training and entropy-constrained pseudo labeling for semi-supervised extractive summarization. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6447–6456.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Kam-Fai Wong, Mingli Wu, and Wenjie Li. 2008. Extractive summarization using supervised and semi-supervised learning. In *Proceedings of the 22nd international conference on computational linguistics (Coling 2008)*, pages 985–992.

Weidi Xu, Haoze Sun, Chao Deng, and Ying Tan. 2017. Variational autoencoder for semi-supervised text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyoung Park. 2021. Gpt3mix: Leveraging large-scale language models for text augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2225–2239.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34:27263–27277.

Haopeng Zhang, Xiao Liu, and Jiawei Zhang. 2023. Extractive summarization via chatgpt for faithful summary generation. *arXiv preprint arXiv:2304.04193*.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

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*, 12:39–57.

Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208.

Yingying Zhuang, Jiecheng Song, Narayanan Sadagopan, and Anurag Beniwal. 2023. Self-supervised pre-training and semi-supervised learning for extractive dialog summarization. In *Companion Proceedings of the ACM Web Conference 2023*, pages 1069–1076.

A Setting T

We experiment with different values of T (number of groups to divide the training set into) and report the validation performance reported in Table 3.

We find that $T = 10$ provides the best trade-off between the number of clusters and model performance as increasing T further leads to minimal gains or sometimes no gain at all.

	TweetSumm		WikiHow	
T	ROUGE-2	L-Eval	ROUGE-2	L-Eval
5	52.1	67.7	6.1	65.3
10	54.3	69.2	7.2	70.2
15	54.2	69.6	7.6	70.5
20	54.4	69.6	7.7	71.1

Table 3: Validation ROUGE-2 and L-Eval scores for different values of T on the TweetSumm datasets.

B Additional Details for *PPSL*.

Computing Confidence. We compute the teacher confidence for a generated summary (a.k.a pseudo-label) as follows: for extractive summarization, and a PreSumm teacher model, let p_{ij} denote the probability with which the i-th sentence s_i in an unlabeled document u_j is present in its summary S_j, and let $\mathbb{1}$ denote the indicator function: $\mathbb{1}(s_i) = \begin{cases} 1, & \text{if } s_i \in S \\ 0, & \text{otherwise} \end{cases}$. We then compute the teacher confidence for the pseudo-label S_j by averaging the probabilities of selected sentences. We define the teacher confidence (C_j) for an input text u_j as follows:

$$C_j = \frac{\sum_{i=1}^{|u_j|} \mathbb{1}(s_i) \cdot p_{ij}}{n}, \tag{2}$$

where $|u_j|$ denotes the number of sentences in the unlabeled document u_j and n is the number of sentences in the generated summary.

Baselines We compare our *PPSL* with the following baselines: **1) PreSumm** (Liu and Lapata, 2019). The original PreSumm model that pretrains a BERT model for summarization. We train two PreSumm models – one on a limited training set with 50 labeled examples to match the starting point of our semi-supervised setting and another with 300 labeled examples, the same as the dataset size at the end of our training cycle. **2) Teacher-Student Learning (TSL)** (Zhuang et al., 2023). Current state-of-the-art semi-supervised method on TweetSumm. The teacher-student learning framework uses a similar formulation for computing model confidence to ours, as follows: $C_j = \sum_{i=1}^{n}(C_{ij})/n_j$. Here, $C_{ij} = p_{ij}q_{ij} + (1-p_{ij})(1-q_{ij})$, where p_{ij} is the probability of sentence i being selected for summary for dialog j estimated by the teacher model, and $q_{ij} = 1$ if p_{ij} in top 4, else 0. We report the performance of the TSL (50:500) and TSL (500:500) models from the paper, as they are the closest to our setup (50/500 labeled examples + 500 unlabeled examples). **3) Confidence + G-4 relabeling + G-4 score (Ours).** Our proposed method following the methodology in Section 4. We first use the PreSumm teacher model to shortlist 50 pseudo-labels (Stage 1 and 2), relabel them using GPT-4 (Stage 3), and then select the top 5 using GPT-4 score (Stage 4). **4) Confidence + G-4 score.** We skip Stage 3 from **3)** to directly score the top 50 PreSumm pseudo-labels using GPT-4. We run this baseline to measure the effect of relabeling in our pipeline. **5) Confidence + G-4 relabeling.** We skip Stage 4 from **3)** and select the final 5 pseudo-labels based on PreSumm confidence. **6) Confidence + L-3 relabeling + L-3 score (Ours).** Same as **3)** but using LLaMA-3. **7) Confidence + L-3 score (Ours).** Same as **4)** but using LLaMA-3. **8) Confidence.** We skip Stage 3 and 4 from from **3)** and select 5 PreSumm pseudo-labels based on PreSumm confidence. **9) Random.** Same as **6)** but instead of using teacher confidence defined in Equation 2, we randomly select five PreSumm pseudo-labels to include in each cycle. The results for these baselines is shown in Table 4.

C Implementation Details

Data Augmentation. We set the number of groups $T = 10$ for all datasets[5] and randomly sample 5 examples from each group to get a 50-shot $\mathcal{D}_{F,train}$. Then, we obtain $\mathcal{D}_{A,train}$ by generating 1000 examples using the procedure described in Section 4. In the data generation prompt, we include five examples for each group for TweetSumm and WikiHow, but for ArXiv/PubMed, we could only fit two documents at a time in LLaMA-3's context window after applying the following truncation heuristic to the text. We include l lines before and after each sentence in the ground truth summary such that we are able to fit two examples in the prompt. The average value of l was 5.21 (so approximately ∼90 for an average summary size of 8 sentences for the ArXiv/PubMed dataset sentences were selected for example[6]). Here, we set the summary size p to 4 sentences for Tweet-

[5] based on validation R-2 and L-Eval scores reported in Table 3 of Appendix A

[6] $(5.21 \times 2 \times 8) + 8 = 91.1$

	TweetSumm			WikiHow			ArXiv/Pubmed		
Method	**R-1 (%)**	**R-2 (%)**	**R-L (%)**	**R-1 (%)**	**R-2 (%)**	**R-L (%)**	**R-1 (%)**	**R-2 (%)**	**R-L (%)**
DistilBERT$_{base}$ (50 labels)									
Random	36.7 (1.5)	25.4 (1.4)	36.7 (1.3)	19.7 (1.4)	1.5 (1.1)	7.2 (1.3)	19.5 (1.3)	2.9 (0.9)	7.8 (1.2)
Confidence	43.5 (1.4)	35.1 (1.2)	46.8 (1.1)	21.3 (0.4)	3.7(0.8)	10.3 (1.0)	23.4 (1.1)	5.2 (0.7)	12.5 (1.1)
+ G-4 relabeling	55.4 (1.3)	**46.7 (0.6)**	56.1 (0.9)	22.1 (0.4)	5.7 (0.6)	13.5 (0.7)	23.8 (0.8)	7.3 (1.3)	15.3 (0.8)
Confidence + G-4 score	46.8 (1.3)	37.4 (0.4)	48.3 (1.2)	21.7 (0.9)	4.6 (0.4)	12.1 (1.1)	24.1 (0.9)	6.7 (0.3)	13.8 (1.4)
+ G-4 relabeling (Ours)	**57.6 (1.2)**	46.3 (1.7)	**56.2 (1.3)**	**22.7 (0.3)**	**5.9 (0.4)**	**13.8 (0.5)**	**24.7 (0.9)**	**8.1 (1.3)**	**15.9 (0.8)**
Confidence + L-3 score	45.7 (1.1)	36.9 (0.2)	47.8 (1.2)	21.6 (0.4)	4.1 (0.5)	11.1 (0.8)	23.9 (0.9)	6.1 (0.3)	12.9 (1.3)
+ L-3 relabeling (Ours)	56.2 (1.1)	45.1 (1.2)	55.9 (1.1)	22.3 (0.1)	5.8 (0.2)	13.6 (0.3)	24.5 (0.6)	7.7 (1.4)	15.7 (0.3)
BERT$_{base}$ (50 labels)									
TSL (50:500)	49.0	37.7	48.2	-	-	-	-	-	-
Random	45.4 (1.4)	32.4 (1.9)	42.5 (1.8)	22.1 (1.7)	2.4 (1.5)	9.6 (1.5)	23.3 (1.4)	6.1 (1.2)	12.4 (1.3)
Confidence	49.7 (1.6)	39.5 (1.4)	49.4 (1.3)	24.5 (0.6)	4.8 (1.1)	12.8 (1.0)	27.6 (1.1)	7.7 (0.7)	14.2 (1.2)
+ G-4 relabeling	57.8 (1.2)	50.3 (0.5)	58.9 (1.2)	**26.4 (0.3)**	**7.3 (0.5)**	**16.4 (0.8)**	28.7 (0.9)	9.5 (1.1)	17.1 (0.8)
Confidence + G-4 score	52.3 (1.6)	42.8 (0.7)	51.0 (1.4)	25.2 (0.7)	5.6 (0.5)	13.1 (1.0)	27.7 (0.9)	7.9 (0.2)	15.5 (1.3)
+ G-4 relabeling (Ours)	**58.9 (1.4)**	**50.4 (0.8)**	**59.4 (1.5)**	26.1 (0.4)	7.2 (0.6)	15.9 (0.9)	**29.1 (0.7)**	**9.7 (1.2)**	**17.7 (0.6)**
Confidence + L-3 score	51.7 (1.2)	41.6 (1.2)	50.3 (1.2)	25.9 (0.3)	5.2 (0.2)	13.0 (0.8)	27.6 (0.4)	7.9 (0.5)	15.3 (1.1)
+ L-3 relabeling (Ours)	58.4 (1.2)	50.1 (0.3)	59.1 (1.2)	26.0 (0.2)	6.9 (0.3)	15.1 (0.2)	29.0 (0.5)	9.4 (0.7)	17.4 (0.3)
BERT$_{base}$ (500 labels)									
TSL (500:500)	59.0	48.3	58.2	-	-	-	-	-	-
Random	55.1 (1.4)	42.7 (1.1)	50.3 (1.2)	25.6 (1.3)	4.5 (1.1)	15.2 (1.3)	25.4 (1.5)	9.5 (1.2)	24.1 (1.2)
Confidence	61.8 (0.7)	54.9 (0.8)	60.3 (0.9)	28.4 (0.6)	8.0 (1.1)	22.5 (1.0)	29.4 (0.5)	11.5 (0.6)	27.7 (0.8)
+ L-3 score	63.4 (0.5)	55.6 (0.8)	62.1 (0.7)	28.9 (0.4)	8.2 (0.5)	28.4 (0.4)	31.7 (0.3)	11.8 (0.2)	29.4 (0.4)
+ L-3 relabeling (Ours)	**64.2 (0.2)**	**56.2 (0.4)**	**62.8 (0.6)**	**30.7 (0.4)**	**8.8 (0.3)**	**29.5 (0.3)**	**33.5 (0.3)**	**12.3 (0.2)**	**32.2 (0.3)**

Table 4: Mean (Std.) ROUGE F-1 scores of different pseudo-labeling strategies. R-1, R-2, and R-L denote ROUGE-1, ROUGE-2, and ROUGE-L metrics, respectively. TSL results from (Zhuang et al., 2023). Refer to Section 5 for method details. **Bold** indicates the best-performing and underline denotes the second-best performing method, respectively.

	TweetSumm		WikiHow		ArXiv/Pubmed	
Method	**R-2**	**L-Eval**	**R-2**	**L-Eval**	**R-2**	**L-Eval**
PreSumm (50 labels)	37.1 (1.1)	31.2 (0.5)	3.2 (0.8)	34.2 (1.5)	7.3 (0.9)	13.5 (1.2)
PreSumm (300 labels)	51.1 (2.1)	60.5 (1.2)	7.6 (0.6)	68.1 (1.1)	10.8 (0.9)	49.5 (2.4)
PreSumm (500 labels)	54.4 (1.2)	67.1 (0.3)	7.9 (0.5)	74.4 (0.6)	11.3 (0.5)	58.2 (1.1)
PreSumm (750 labels)	56.1 (0.7)	70.3 (0.5)	8.5 (0.4)	76.5 (0.4)	12.1 (0.7)	62.8 (0.7)
50 labels						
Random	32.4 (1.9)	32.1 (1.1)	2.4 (1.5)	37.7 (1.6)	6.1 (0.2)	15.1 (2.3)
Confidence + G-4 score	42.8 (0.7)	46.2 (0.2)	5.6 (0.5)	59.4 (1.3)	7.9 (0.2)	40.1 (1.9)
+ G-4 relabeling (Ours)	50.4 (0.8)	58.4 (0.4)	7.2 (0.6)	70.3 (1.4)	9.7 (1.2)	52.5 (1.3)
Confidence + L-3 score	41.6 (1.2)	45.8 (0.7)	5.2 (0.2)	57.5 (1.4)	7.9 (0.5)	37.1 (1.8)
+ L-3 relabeling (Ours)	50.1 (0.3)	56.3 (0.9)	6.9 (0.3)	69.3 (2.1)	9.4 (0.7)	49.3 (1.4)
500 labels						
Random	42.7 (1.1)	52.3 (1.2)	4.5 (1.1)	52.7 (1.8)	9.5 (1.2)	44.1 (0.9)
Confidence + L-3 score	55.6 (0.8)	69.2 (0.7)	8.2 (0.5)	75.2 (1.4)	11.8 (0.2)	60.2 (0.5)
+ L-3 relabeling (Ours)	**56.2 (0.8)**	**71.2 (0.9)**	**8.8 (0.3)**	**77.3 (1.3)**	**12.3 (0.2)**	**65.7 (0.3)**

Table 5: **Fully-supervised methods (first four rows) semi-supervised approaches (remaining rows).** All models use BERT$_{base}$ as PreSumm's backbone. The number of labeled examples for fully supervised models is shown in brackets. The semi-supervised methods use 50/500 labeled and 250 unlabeled examples.

Summ and WikiHow datasets, and 8 sentences for the Arxiv/PubMed dataset. We determine these summary sizes based on the average summary size in the few-shot training data $\mathcal{D}_{F,train}$. We host `LLaMA-3-70b-Instruct` on 4×A100 GPUs with 80G VRAM each and use it as the backbone LLM for all our experiments. Generating $\mathcal{D}_{A,train}$ took ∼4.2 hrs for TweetSumm, ∼11.3 hrs for WikiHow, and ∼1.4 days for ArXiv/PubMed dataset.

Training. For extractive summarization, we train a PreSumm model on the combined *MixSumm*-generated and seed few-shot dataset $\mathcal{D}_{F+A,train}$. We use the TransformerSum repository[7] to implement our training pipeline. To handle long documents that cannot be fed to the PreSumm at once, we introduce a subroutine that iteratively chunks and summarizes the document until we obtain a summary of size p. The iterative subroutine is crucial to train PreSumm models on the WikiHow and ArXiv/PubMed datasets with long input documents. For abstractive summarization, we follow an *extractive-then-abstractive* approach, where for a given input document, we first obtain its extractive summary using the full-trained PreSumm model from the previous step. Then, we finetune a DistilBART model that summarizes the PreSumm summaries to generate abstractive summaries.

We initialize the training process with a learning rate of 2×10^{-5} and use a cyclic learning rate scheduler (Smith, 2015). We train all our models for 100 epochs with an early stopping criterion, where we

[7] `https://transformersum.readthedocs.io/`

stop the training process if the validation ROUGE-2 score does not improve for more than 10 epochs. We use the AdamW optimizer (Loshchilov and Hutter, 2017) with $\epsilon = 1 \times 10^{-8}, \beta_1 = 0.9, \beta_2 = 0.99$ and train all our models on one V100 GPU with 12G VRAM. We use `distilbart-12-6-cnn` backbone for abstractive summarization and experiment with two backbones for the PreSumm model: DistilBERT$_{base}$ and BERT$_{base}$ (results in Table 6) and find BERT$_{base}$ to be better. Training a model on *MixSumm*-generated data took ∼2.5 hrs for TweetSumm, ∼13.4 hrs, for WikiHow, and ∼2.7 days for ArXiv/PubMed. Crucially, we repeat each experiment (data augmentation+model training) for 5 random seeds and report the mean and standard deviations for all models unless otherwise stated.

Semi-Supervised Text Summarization. We use the TransformerSum repository[8] to implement our training pipeline. We use PreSumm as our teacher model M and experiment with two backbones: `distilbert-base-uncased` and `bert-base-uncased`. We perform experiments in two settings: **1)** data-scarce setting where fix the size of the labeled set D_l to 50 for all the datasets, and **2)** data-abundant setting where we set the size of D_l to 500. We set N_{cycles} to 50 for all experiments. We add 5 pseudo-labels to the training set in each cycle, thus resulting in a final training set size of 300 (50 labeled + 250 pseudo-labeled examples.) We set the summary size k to 4 for TweetSumm and 8 for WikiHow and ArXiv/PubMed. We base these summary sizes on the average summary size of the labeled training set. For training, we start with a learning rate of 2×10^{-5} on all the datasets and use a cyclic learning rate scheduler during training (Smith, 2015), which is the default setting in TransformerSum. Additionally, we use AdamW as our optimizer with $\epsilon = 1 \times 10^{-8}, \beta_1 = 0.9, \beta_2 = 0.99$. We train all our models on a single V100 GPU with 12G VRAM. We repeat each experiment for three different seeds and report the mean and standard deviation in our results unless otherwise stated.

D Qualitative Results

D.1 On the Sample Efficiency of *PPSL*

We now compare the sample efficiency of *PPSL* against other methods. Referring to Table 5, for fully supervised methods, we note that including more labeled examples improves L-Eval and ROUGE scores across the board (“PreSumm (50 labels)" v/s “PreSumm (300 labels)"). Our semi-supervised approach using 50 labels with GPT-4 relabeling and GPT-4 score achieves competitive performance to the fully supervised PreSumm model trained on 300 labels. Notably, we get better L-Eval scores than “PreSumm (300 labels)" on WikiHow and ArXiv/PubMed datasets and are competitive on TweetSumm. Note that the “PreSumm (300 labels)" model approximates the best-case scenario when all the labels in the training set are high-quality. This is encouraging, as “PreSumm (300 labels)" approximates the best-case scenario of 100% high-quality labels in the training set. In the data-abundant setting, our proposed method with LLaMA outperforms the respective fully supervised model in terms of both ROUGE and L-Eval. From Table 2, we further note that our approach outperforms TSL (50:500) while using half the number of pseudo-labels. We may further improve the model performance by including some examples in the prompt. Our proposed method outperforms TSL (50:500) and TSL (500:500) despite working in a more challenging labeled:unlabeled dataset ratio of 50:250. We plot the R-1 scores against the number of training cycles for *PPSL* and other semi-supervised baselines (refer to Section B in Appendix B for more details) in Figure 4. Overall, “Random" setting is highly unstable, “Confidence + G-4 score" slightly improves over “Confidence" on TweetSumm and WikiHow, but more importantly, it is consistently more stable. Finally, our method with GPT-4 scoring and relabeling not only significantly boosts the R-1 scores (visible gap between “Ours" and the rest) but also does so at a much faster rate. For all the datasets, our method peaks and stabilizes under 20 cycles (100 pseudo-labels), further endorsing the sample efficiency of our method compared to other approaches.

D.2 Comparison of Pseudo-label Selection Strategies

Referring to Table 2, we note that all pseudo-label selection strategies outperform the random baseline. The “Random" baseline performs worse than the fully supervised counterpart on all datasets (R-2 in Table 2 v/s R-2 in Table 5), meaning that the *majority* of the shortlisted PreSumm pseudo-labels are low-quality. Using teacher confidence leads to slight performance gains on all the datasets, and

[8] `https://transformersum.readthedocs.io/en/latest/`

	TweetSumm				**WikiHow**				**ArXiv/Pubmed**			
Method	**R-1 (%)**	**R-2 (%)**	**R-L (%)**	**L-Eval (%)**	**R-1 (%)**	**R-2 (%)**	**R-L (%)**	**L-Eval (%)**	**R-1 (%)**	**R-2 (%)**	**R-L (%)**	**L-Eval (%)**
DistilBERT$_{base}$												
Oracle	*62.8*±0.6	*53.1*±1.2	*59.3*±0.7	*83.6*±0.5	*30.7*±0.4	*8.6*±0.8	*19.1*±0.7	*81.6*±1.2	*34.2*±0.6	*12.3*±1.2	*19.4*±0.4	*71.1*±0.4
PPSL (50:250)	56.2±1.1	45.1±1.2	55.9±1.1	-	22.3±0.1	5.8±0.2	13.6±0.3	-	24.5±0.6	7.7±1.4	15.7±0.3	-
EDA	47.3±1.3	36.1±1.2	48.7±1.2	51.3±0.3	21.6±0.8	3.3±0.8	11.8±1.2	54.1±0.3	23.3±1.3	5.4±0.6	12.6±1.3	39.6±0.2
MixSumm (rand.)	56.9±2.5	46.1±3.4	58.7±3.1	56.7±0.6	22.7±2.1	6.1±1.2	14.8±1.3	65.9±0.4	24.8±1.5	8.3±1.7	16.0±1.3	48.1±0.6
w/o Aug.	41.7±1.6	32.4±1.2	43.6±2.1	23.4±1.2	19.2±1.8	2.1±0.6	9.1±1.4	20.4±1.0	21.4±1.2	4.7±0.3	10.4±1.2	13.3±0.5
MixSumm (ours)	**57.3**±2.4	**46.8**±3.1	**57.2**±2.7	**60.3**±0.5	**23.4**±1.7	**6.5**±1.6	**15.2**±1.1	**68.4**±1.3	**25.7**±1.7	**8.6**±2.1	**16.6**±1.4	**51.2**±0.6
w/o Mixup	54.2±1.7	44.3±1.4	53.5±1.4	55.3±1.2	22.1±1.3	4.7±0.2	12.8±1.2	62.3±0.7	23.8±1.2	6.1±0.9	14.1±1.3	42.1±1.1
w/o Aug.	42.8±1.1	34.1±1.1	44.2±1.4	28.4±0.8	19.7±1.2	2.8±0.4	10.2±1.1	31.4±0.4	22.6±1.3	4.9±0.6	11.3±1.3	18.6±0.5
BERT$_{base}$												
Oracle	*65.7*±0.3	*56.6*±0.4	*64.9*±0.4	*86.2*±0.3	*30.5*±0.5	*8.7*±0.3	*19.2*±0.6	*87.3*±0.1	*34.6*±0.4	*12.4*±0.2	*19.6*±0.4	*78.1*±0.5
TSL (50:500)	49.0	37.7	48.2	-	-	-	-	-	-	-	-	-
TSL (500:500)	59.0	48.3	58.2	-	-	-	-	-	-	-	-	-
EDA	51.1±0.7	39.2±0.9	53.0±0.2	34.3±1.2	23.4±0.5	4.1±0.3	13.0±0.5	42.1±0.8	26.2±1.1	7.9±1.0	13.1±0.6	17.2±0.5
PPSL (50:250)	58.4±1.2	50.1±0.3	59.1±1.2	56.3±0.9	26.0±0.2	6.9±0.3	15.1±0.2	69.3±2.1	29.0±0.5	9.4±0.7	17.4±0.3	49.3±1.4
MixSumm (rand.)	58.6±3.2	50.6±2.1	59.7±2.3	60.3±0.9	26.4±1.0	7.5±1.2	15.8±0.2	72.5±1.2	30.7±1.7	**10.6**±1.5	**18.5**±1.1	48.4±1.1
w/o Aug.	49.4±0.7	36.9±1.0	49.0±0.2	31.5±0.5	21.3±0.4	3.2±0.4	11.4±0.5	34.2±1.5	23.4±1.1	7.5±1.4	12.3±0.8	13.5±1.2
MixSumm (ours)	**59.1**±1.7	**52.7**±1.6	**60.5**±1.3	**65.3**±1.2	**27.3**±2.1	**7.8**±1.3	**16.6**±1.8	**81.1**±1.7	**31.2**±1.2	**10.7**±1.1	**18.3**±1.1	**53.1**±0.5
w/o Mixup	56.1±1.1	47.3±1.2	55.3±1.1	57.3±0.5	25.7±1.4	6.2±1.2	14.7±0.7	67.3±2.1	28.4±1.9	8.3±1.3	16.8±1.6	52.3±1.2
w/o Aug.	50.1±0.6	38.1±1.0	49.9±0.6	32.3±3.1	21.9±0.3	3.5±0.2	12.1±0.9	33.3±1.7	24.1±0.9	7.9±1.0	12.7±0.5	19.0±2.5
LLaMA-3 (0-shot)	50.3±0.5	47.7±0.4	49.9±0.3	52.3±1.2	12.2±0.2	2.7±0.5	8.1±0.4	32.3±0.3	23.6±0.2	4.6±0.7	15.4±0.3	**38.4**±0.5
LLaMA-3 (1-shot)	51.7±0.2	49.2±0.3	51.9±0.3	58.7±1.1	14.3±0.2	4.1±0.5	10.6±0.2	39.4±0.5	**32.6**±0.4	**6.5**±0.7	**17.2**±0.3	38.3±1.8
LLaMA-3 (5-shot)	**62.4**±0.5	**54.3**±0.7	**60.3**±1.1	**67.5**±0.6	**28.7**±0.3	**7.5**±0.9	**17.1**±0.3	**71.3**±0.4	-	-	-	-

Table 6: **Extractive Summarization Results.** Comparison of different text summarization models on TweetSumm, WikiHow, and ArXiv/PubMed datasets. We report ROUGE-1 (R-1), ROUGE-2 (R-2), ROUGE-L (R-L) F_1 scores, and L-Eval scores. We report the mean±std. performance across 5 different seeds. Refer to Appendix C and Section 5.3 for metric and implementation details. **Note.** TSL results are reported from Zhuang et al. (2023). For EDA and *MixSumm* we use a 50-shot $\mathcal{D}_{F,train}$ and generate 1000 examples as $\mathcal{D}_{A,train}$. **Bold** denotes the best-performing model in a given block and highlight denotes the *overall* best-performing model. For the ArXiv/PubMed dataset, we could fit only 2 documents into LLaMA-3's context (1 from $\mathcal{D}_{F,train}$ + 1 generated), so we do not report LLaMA-3 (5-shot).

adding GPT-4 score further improves the results ("Confidence" v/s "Confidence + G-4 score" in Table 2). These improvements indicate that the shortlisted PreSumm pseudo-labels include some good-quality pseudo-labels, too, and using GPT-4 to rate those pseudo-labels is crucial to picking them. We see similar trends when using LLaMA-3.

To further confirm our findings, we conduct a qualitative study in the data-scarce setup, where we compute the ROUGE scores of the final 5 pseudo-labels for each method against the respective ground truth summaries, and Figure 5 shows the mean ROUGE-2 of the five selected pseudo-labels. To clarify, we obtained the "Oracle" results by directly selecting the final 5 pseudo-labels using ROUGE-2 scores computed against the ground truth. We note a stark difference between "Confidence" and "Oracle," which shows that relying solely on teacher confidence consistently leads to a selection of low-quality pseudo-labels. Combining GPT-4 score with teacher confidence is effective ("Confidence + G-4 score"), and adding the GPT-4 relabeling greatly boosts the quality of selected pseudo-labels ("Ours").

D.3 Effect of Relabeling

Referring to Tables 2 and 5, we observe that relabeling with LLMs leads to a significant boost in the summarization performance in terms of both ROUGE scores and L-Eval. When using BERT$_{base}$ as the backbone, we note that the ROUGE-1 improves from 52.3 to 58.9 on the TweetSumm dataset, 25.2 to 26.1 on the WikiHow dataset, and 27.7 to 29.1 on the ArXiv/PubMed dataset. GPT-4 relabeling is also effective when using teacher confidence without GPT-4 score ("Confidence" v/s "Confidence + G-4 relabeling"). Our previous qualitative study also supports these results, showing that relabeling improves the quality of pseudo-labels. We observe similar trends when using DistilBERT$_{base}$ as PreSumm's backbone and LLaMA-3 instead of GPT-4. When using 500 labels, we note boosts in performance but the relative scale is smaller compared to when using 50 labels.

We conduct additional testing to analyze the performance of our best- and second-best-performing models, both of which involve relabeling. We find that the p-value < 0.016 for Welch's test for an R-1 of 58.9 (1.4) for "Confidence + G-4 score + G-4 relabelling" v/s 57.8 (1.2) for "Confidence + G-4

EDA
Generated Document: Customer:@VerizonSupport I penury need help. I do not have your service (6+ iodine hours to setup and cancel an install!) but the premature homeowner did. There is an ONT in the basement that every out a loud, very annoying beep sends 15 minutes? This is ridiculous. How do I iodine stop it? Agent:@633638 Press and hold the alarm silence button/ONT reboot button on the bring up unit for 30 seconds. That bequeath will silence it. Customer:@VerizonSupport done. Will reply back respond later if it beeps again. @VerizonSupport comprise I expected to do this every 24 hours? That's how long the Alarm Silence lasted. Bedden back to beeping every 15 moment since this morning. Agent:@633638 assay unplugging the power and removing the battery. indium After 3mins, put the battery back in and power on. This should assistant with the beeping. Let the states know. D̂DG Customer:@VerizonSupport I no service with you at all. Can't I just unplug it and remove assault and battery the battery forever? Agent:@633638 Yes, you can do that since you do not inspection and repair have service with us. personify When the battery is dead the beeping will stop. ĤSB Customer:@VerizonSupport you! Thank Agent:@633638 personify You are welcome!
LLaMA-3 Summary: Customer:@VerizonSupport I penury need help. There is an ONT in the basement that every out a loud, very annoying beep sends 15 minutes? This is ridiculous. How do I iodine stop it? Agent:@633638 Press and hold the alarm silence button/ONT reboot button on the bring up unit for 30 seconds. That bequeath will silence it. Customer:@VerizonSupport done. Will reply back respond later if it beeps again. @VerizonSupport comprise I expected to do this every 24 hours? Agent:@633638 assay unplugging the power and removing the battery.
***MixSumm* w/o mixup**
Generated Document: Customer: My iPhone's battery is draining much faster than usual, and it's overheating. Any suggestions? Here's a link to a screenshot of my battery usage: https://t.co/xyzABC123 @AppleSupport can you help with this issue? Agent: @UserHelp123 We're here to assist you. First, could you let us know which iOS version your iPhone is currently using? Customer: @AppleSupport It's updated to iOS 14.4. Agent: @UserHelp123 Thanks for the information. Have you installed any new apps or updates around the time you started noticing these issues? Customer: @AppleSupport Yes, I installed a few new games last week. Agent: @UserHelp123 It's possible that those games are causing the battery drain and overheating. Could you try removing them to see if the situation improves? Customer: @AppleSupport I've uninstalled the games, but my phone is still overheating. Agent: @UserHelp123 Understood. Let's perform a soft reset on your iPhone. This won't erase your data but can help with such issues. If the problem persists, we should continue this conversation in DM to get more details and possibly set up a service appointment. Please reach out in DM with the exact time you started experiencing these issues and if you've noticed any specific patterns. https://t.co/NmDpIWxYfO
LLaMA-3 Summary: Customer: My iPhone's battery is draining much faster than usual, and it's overheating. Any suggestions? Agent: Have you installed any new apps or updates around the time you started noticing these issues? Customer: @AppleSupport I've uninstalled the games, but my phone is still overheating. Agent: Let's perform a soft reset on your iPhone. This won't erase your data but can help with such issues.
MixSumm
Generated Document: Customer:@VerizonSupport I recently moved into a house that has your equipment but I don't use your service. The device in the basement beeps every 20 minutes. How do I stop it? Agent:@VerizonHelper Please press and hold the alarm silence button on the ONT for about 30 seconds. That should silence it temporarily. Customer:@VerizonSupport I did that, but it started beeping again after a few hours. I need a permanent fix. I have a flight next week, and I can't deal with this noise. Agent:@VerizonHelper If the alarm silence is temporary, try unplugging the device and removing its battery. After a few minutes, reconnect the battery and power it on. Customer:@VerizonSupport I need this sorted out before I leave. Is there no faster way? Agent:@VerizonHelper For a quick solution, you can permanently disconnect the power and remove the battery. This will stop the beeping entirely. Customer:@VerizonSupport Thank you for the advice! I'll handle it immediately.
LLaMA-3 Summary: Customer:@VerizonSupport I recently moved into a house that has your equipment but I don't use your service. The a device in the basement beeps every 20 minutes. How do I stop it? Agent:@VerizonHelper Please press and hold the alarm silence button on the ONT for about 30 seconds. That should silence it temporarily. Customer:@VerizonSupport I did that, but it started beeping again after a few hours. I need a permanent fix. I have a flight next week, and I can't deal with this noise. Agent:@VerizonHelper For a quick solution, you can permanently disconnect the power and remove the battery. This will stop the beeping entirely.

Table 7: Full qualitative comparison of EDA, *MixSumm* w/o mixup, and *MixSumm* examples. For each method, the first row shows the generated document and the second row shows its LLaMA-3-generated summary. For these examples, group 1 contained customer conversations with phone companies, such as Verizon, and group 2 contained customer interactions with airline representatives, such as AirAsia and Delta. **Note.** For brevity, we do not include the input examples used in the prompt, and for EDA, we perform augmentations one sentence in the document at a time. We note that w/o mixup, *MixSumm* generated a decent quality document but it only covers phone/electronic device-related sentences. *MixSumm*, on the other hand, generated an example that contains mention of flight as well as a device-related issue. Additionally, we note that no matter the quality of the document, LLaMA-3-70b generates a high-quality summary in all cases.

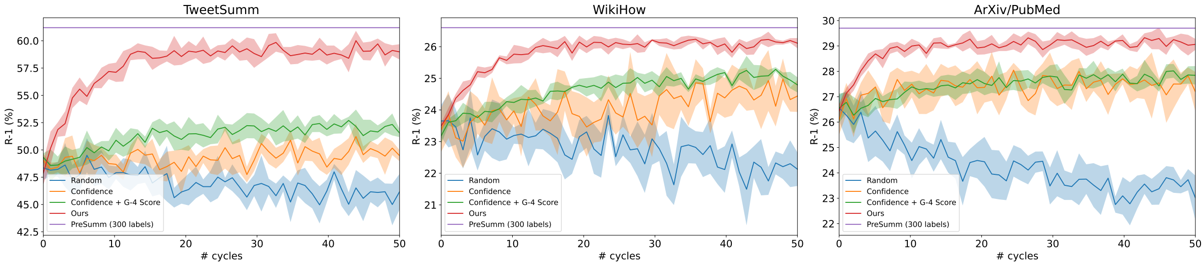


Figure 4: **ROUGE-1 curves v/s # cycles for data-scarce setting.** Each cycle denotes an addition of 5 new pseudo-labels to the training set. All results use BERT$_{base}$ as the backbone for PreSumm. The curves are averaged for three seeds (the width denotes the std). Note that we report the GPT-4 version of our method here.

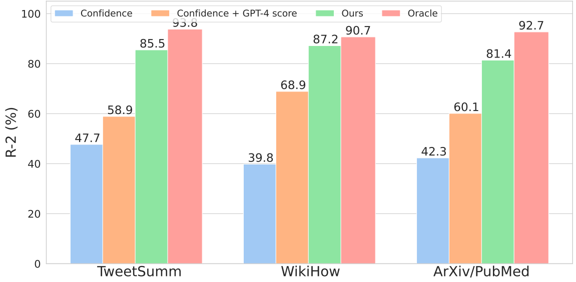


Figure 5: **Quality of pseudo-labels by different strategies (data-scarce setup).** The y-axis denotes the ROUGE-2 scores of the top 5 pseudo-labels computed against the respective ground truths. All results are for BERT$_{base}$ as the backbone for PreSumm and three random seeds. Refer to Section D.2 for complete details.

relabeling" on the TweetSumm dataset, denoting the differences are significant.

E Prompt Designs

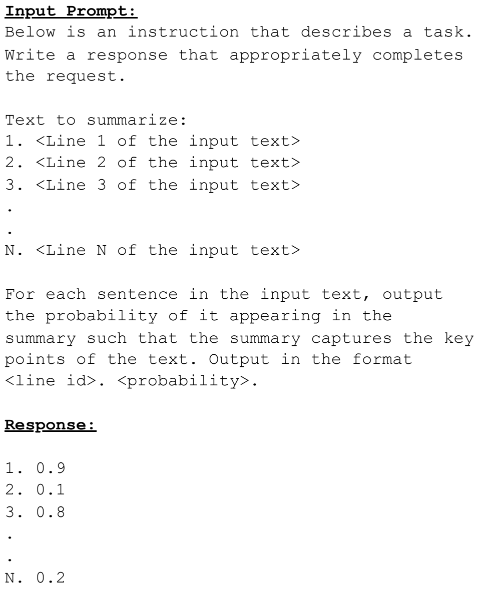
In this section, we show our prompts to synthesize new documents and their summaries.

Prompt 1: Prompt used for Generating New Articles

```
### Instruction:
You are an expert data generator tasked with synthesizing new documents for a
summarization task. {dataset_description}

Below are some example documents and their summaries from a group in the dataset
(group 1):
{gp1_documents}

Below are some example documents and their summaries from another group in the
    dataset (group 2):
{gp2_documents}

Given the above documents, follow these instructions:
* Synthesize a new document that follows a similar format to the examples provided.
* The document should contain {document_size}.
* The document should be coherent and relevant to the topic.
* The document should be original and not copied from the examples.
* Ensure that the document covers {alpha}% topics from group 1 and 1 - {alpha}%
    topics from group 2.
* Wrap your response in the <document></document> tags.
* Do NOT include anything else like the examples in your output.

### Response:
```

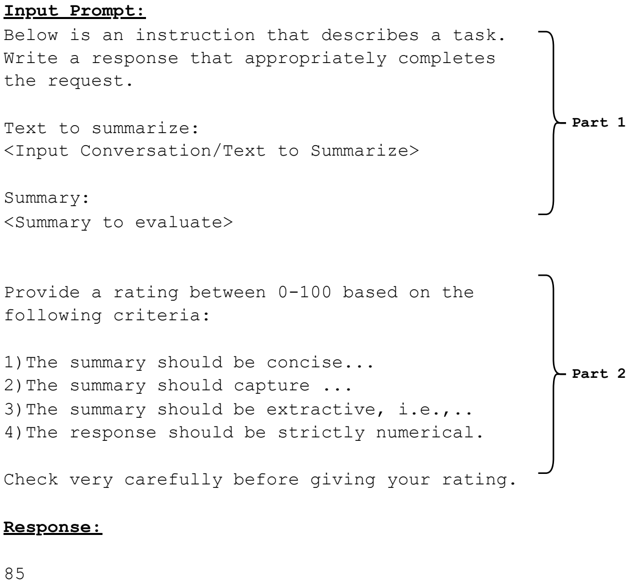
Prompt 2: Prompt used for Scoring a Generated Summary

```
Below is an instruction that describes a task. Write a response that appropriately
completes the request.

### Instruction:
Given the document:
{document}

Provided Summary:
{summary}

Follow these instructions when writing your response:
* On a scale of 1-10, provide a numerical rating for the provided summary, with 10
    denoting that the provided answer perfectly surmises the main points of the
    document.
* Your response should contain only the numerical rating. DO NOT include anything
    else like the provided answer, the ground truth answer, or an explanation of
    your rating scale in your response.
* Wrap your numerical rating inside <rating></rating> tags.
* Check very carefully before answering.
* Follow the output format as shown in the example below:
Example response:
<rating>7</rating>

### Response:
```



```
Input Prompt:
Below is an instruction that describes a task.
Write a response that appropriately completes
the request.

Text to summarize:
1. <Line 1 of the input text>
2. <Line 2 of the input text>
3. <Line 3 of the input text>
.
.
N. <Line N of the input text>

For each sentence in the input text, output
the probability of it appearing in the
summary such that the summary captures the key
points of the text. Output in the format
<line id>. <probability>.

Response:

1. 0.9
2. 0.1
3. 0.8
.
.
N. 0.2
```

(a) **Generating pseudo-labels.** We attach a line ID to each sentence in the input document and instruct the LLM to use those line IDs in its response.



```
Input Prompt:
Below is an instruction that describes a task.
Write a response that appropriately completes
the request.

Text to summarize:                                  Part 1
<Input Conversation/Text to Summarize>

Summary:
<Summary to evaluate>


Provide a rating between 0-100 based on the
following criteria:

1)The summary should be concise...
2)The summary should capture ...                    Part 2
3)The summary should be extractive, i.e.,..
4)The response should be strictly numerical.

Check very carefully before giving your rating.

Response:

85
```

(b) **Scoring pseudo-labels.** The two-part prompt contains a text and summary pair (Part 1), and a list the evaluation criteria (Part 2). **Note:** Refer to Section 4.2 for complete details on the evaluation criteria.

Figure 6: Different prompts used in the experiments.

Prompt 3: Prompt used for Summarizing an Article in *MixSumm*

```
Below is an instruction that describes a task. Write a response that appropriately
completes the request.

### Instruction:
You are an expert data annotator tasked with summarizing documents for a
    summarization task. {dataset_description}

Below are some example documents and their summaries from a group in the dataset (
    group 1):
{gp1_documents}

Below are some example documents and their summaries from another group in the
    dataset (group 2):
{gp2_documents}

A document is composed of the following sentences:
{sentences}

Given the sentences above:
* You are to construct an extractive summary for the document by selecting some
    sentences from above.
* The summary captures the main points of the article.
* Now, output the probability of a sentence being included in the summary.
* Do NOT include anything else like the sentence in your output.
* Output your probabilities in the format <line id>. <probability>. Refer to the
    example below:
1. 0.73
2. 0.65
3. 0.95
etc.

### Response:
```