Domain Aligned Prefix Averaging for Domain Generalization in Abstractive Summarization

Pranav Ajit Nair[1], **Sukomal Pal**[1], **Pradeepika Verma**[2]

[1]Indian Institute of Technology (BHU), Varanasi, India
[2]TIH, Indian Institute of Technology, Patna, India
[1]{pranavajitnair.cse18, spal.cse}@itbhu.ac.in
[2]pradeepikav.verma093@gmail.com

Abstract

Domain generalization is hitherto an underexplored area applied in abstractive summarization. Moreover, most existing works on domain generalization have sophisticated training algorithms. In this paper, we propose a lightweight, weight averaging based, **D**omain **A**ligned **P**refix **A**veraging approach to domain generalization for abstractive summarization. Given a number of source domains, our method first trains a prefix for each one of them. These source prefixes generate summaries for a small number of target domain documents. The similarity of the generated summaries to their corresponding documents is used for calculating weights required to average source prefixes. In DAPA, prefix tuning allows for lightweight finetuning, and weight averaging allows for the computationally efficient addition of new source domains. When evaluated on four diverse summarization domains, DAPA shows comparable or better performance against the baselines, demonstrating the effectiveness of its prefix averaging scheme[1].

1 Introduction

Abstractive document summarization aims at filtering the most crucial information in a document to present a concise view of it (Nallapati et al., 2016; See et al., 2017). This document may take the form of a news article (Hermann et al., 2015), a scientific paper (Yasunaga et al., 2019), a dialogue (Gliwa et al., 2019), or a social media post (Kim et al., 2018). The advent of pretrained models (Raffel et al., 2020; Lewis et al., 2020) has significantly improved abstractive summarization on several of the aforementioned domains. However, these approaches require extensive manual labelling of data which limits their use to domains without any labelled data. Given that real-world applications of summarization often face the problem of adapting to new domains, it becomes crucial to develop summarization systems that do well in data-scarce settings by leveraging information from the source domains.

Domain generalization accounts for learning a robust model for unseen domains from a set of source domains. This problem is closely related to transfer learning, multitask learning and domain adaptation all of which involve learning a model from a set of source tasks/domains to perform well on a set of target tasks/domains. However, in the case of domain generalization, labelled data for the target domain is unavailable. Previous works on domain generalization mainly focus on learning domain invariant features (Gulrajani and Lopez-Paz, 2020; Wang, 2020; Li et al., 2018). Such methods work well on classification tasks where learning domain invariant features is sufficient to predict target classes. However, they may be insufficient for language generation tasks which have writing style, grammar as their ingredients (Vu et al., 2022). Moreover, such methods involve sophisticated algorithms for training and cannot be used for lightweight domain generalization.

Prefix tuning (Li and Liang, 2021) is a lightweight approach to adapting pretrained language models to downstream tasks. It augments the transformer self attention via prefix tokens learned through backpropagation on the task data while keeping the pretrained model's parameters frozen. Prompt tuning based approaches have been shown to do well on lifelong learning (Qin and Joty, 2021) and zero-shot domain adaptation (Zhao et al., 2022), which inspires us to adapt it to domain generalization for abstractive summarization. Concurrently, weight averaging has performed well on domain generalization tasks in Computer Vision (Cha et al., 2021; Ramé et al., 2022; Arpit et al., 2021). To improve functional diversity, these methods average model parameters from different runs and/or checkpoints.

Matena and Raffel (2021) applied weight aver-

[1]https://github.com/pranavajitnair/DAPA

Findings of the Association for Computational Linguistics: ACL 2023, pages 4696–4710

aging to NLP tasks. They merged models trained on different tasks/domains through fisher weight averaging. Their promising results motivate us to apply model merging through weight averaging for domain generalization. Keeping in mind the goal of generating a lightweight and parameter-efficient approach, and the benefits brought by weight averaging, we propose a lightweight **D**omain **A**ligned **P**refix **A**veraging, DAPA, approach to domain generalization for abstractive summarization. Our algorithm consists of three stages. First, prefixes are trained for each source domain. In the second stage, these source prefixes generate summaries for a small number of unlabelled target domain documents. In the third stage, the target domain prefix is obtained through a weighted average of these source prefixes. A higher document-summary similarity score, calculated from the summaries generated in the second stage, would assign a greater weightage to the corresponding source prefix. Through our prefix averaging scheme, we can identify source prefixes essential to ensure good performance on the target domain. Our extensive experimentation on four domains demonstrates the benefits bought by DAPA.

DAPA comes with the following advantages: i) It is a lightweight approach to domain generalization since the backbone pretrained model is frozen and only source prefixes are trained. ii) Through our novel prefix averaging scheme, DAPA is able to generalize well onto target domains. Moreover, freezing the backbone model's parameters further preserves generalization. iii) Our approach supports the efficient addition of new source domains since it only involves recomputing the prefix averaging weights.

To this end, we summarize our contributions as follows:

- To the best of our knowledge, we are the first to explore prefix averaging for domain generalization on a language generation task.
- We propose a lightweight **D**omain **A**ligned **P**refix **A**veraging, DAPA, approach to domain generalization for abstractive summarization. DAPA first trains prefixes for each source domain, following which it utilizes the summary generation capabilities of these source prefixes to generalize to the target domain.
- Through our experimentation setup we demonstrate the effectiveness of DAPA on domain generalization for abstractive summarization.

The rest of the paper is structured as follows: We explore related works in Section 2. Section 3 describes our proposed approach DAPA. Section 4 and 5 provide results for our domain generation experiments and a set of analysis we conduct on our approach. Section 6 provides the conclusion and thoughts for future work. Section 7 discusses limitations and Section 8 sheds light om ethical risks of our work.

2 Related Work

2.1 Abstractive Summarization

Abstractive document summarization aims to distill the most critical information in a document to present a concise view of it. Nallapati et al. (2016) used an RNN based sequence to sequence model for abstractive summarization. See et al. (2017) used pointer generator networks to copy words from the input document. Duan et al. (2019) augmented the transformer architecture with a contrastive attention mechanism to ignore the irrelevant parts of the document. Zhang et al. (2020) pretrained a transformer model for summarization. Liu and Liu (2021) used contrastive loss for better re-ranking of summaries generated by pretrained models. Paulus et al. (2018) proposed the use of policy gradient reinforcement learning to alleviate exposure bias. Gehrmann et al. (2018) developed a bottom-up copy attention mechanism to overdetermine phrases in the document that should be included in the summary. Although great progress has been made in advancing state-of-the-art, few works have explored domain generalization for abstractive summarization. In this work, we develop a lightweight prefix averaging based method for domain generalization in abstractive summarization.

2.2 Prompt Learning in Language Generation

Prompts are task specific instructions prepended to the pretrained model's input. These task specific instructions are trained on the downstream task data while keeping the pretrained model's parameters frozen. Li and Liang (2021) proposed deep continuous prefixes that are prepended to the self attention layers of transformers. They demonstrated the effectiveness of using prefixes for language generation tasks such as summarization. Qin and Joty (2021) prepended prompts into model embeddings for lifelong learning on language generation tasks. Similar to our approach, they train

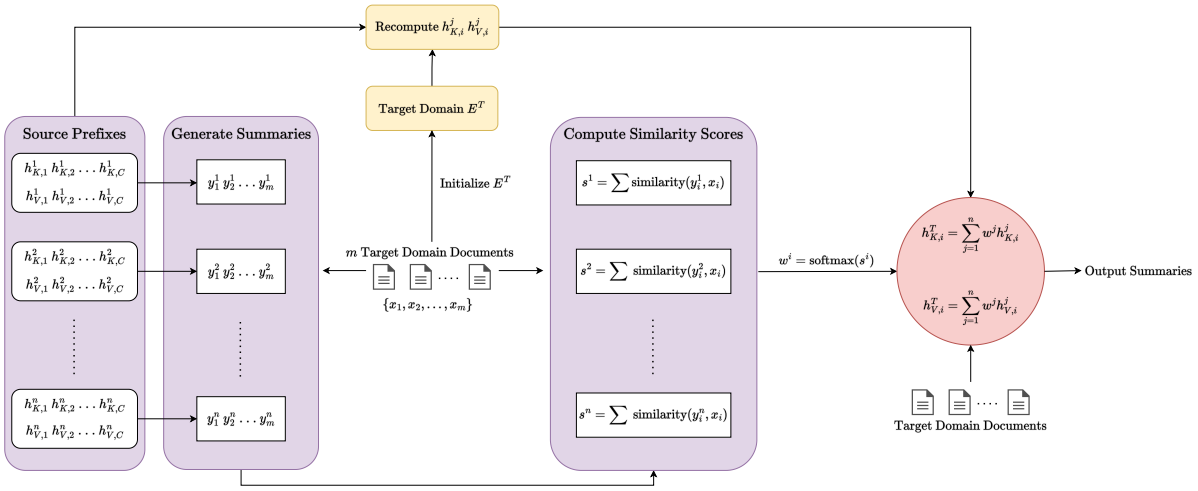


Figure 1: Overview of the Domain Aligned Prefix Averaging model. Source prefixes are used for generating summaries for m target domain documents. The similarity of these summaries to their corresponding documents are used for computing weights required to average source prefixes.

a separate prompt for each domain, however, their work does not focus on domain generalization. Tan et al. (2021) developed a multistage prompting network for machine translation where the encoder is prompted twice to refine the input representation, and the decoder is prompted once to generate the translation. Schick and Schütze (2020) used manually crafted templates for fixed-prompt tuning of pretrained models for few-shot summarization. Zhao et al. (2021); Dou et al. (2021) used learnable prompts as guiding instructions for summarization. Zhao et al. (2022), similar to our approach, used prefixes to adapt to target domain. However, their approach involves pretraining prefix weights and cannot easily incorporate new source domains. On the other hand, our work takes a weighted average of source prefixes and can easily add new source domains to the mix.

2.3 Weight Averaging and Domain Generalization

Domain generalization accounts for learning a robust model for unseen domains from a set of source domains. It has mostly been studied in Computer Vision. The main approaches are based on invariant feature learning (Li et al., 2018; Wang, 2020), data augmentation (Wang et al., 2020a), and meta learning (Wang et al., 2020b). For language generation, Vu et al. (2022) proposed a leave-one-domain-out strategy to fuse adaptors for machine translation. Recently, weight averaging (Garipov et al., 2018) has been successfully applied to domain generalization. Cha et al. (2021) utilized weight averaging to attain a flat minima, and thereby attained generalization across domains. Ramé et al. (2022) averaged weights across multiple runs, trained with different hyperparameters to improve domain generalization. Arpit et al. (2021) ensembled a moving average of weights across multiple checkpoints. Unlike these approaches, we develop a novel mechanism to generate weights for averaging source prefixes, and evaluate our approach on abstractive summarization. Matena and Raffel (2021) also developed a scheme to average model weights. They utilized Fisher information in model parameters for weight averaging. However, they only evaluated on NLU tasks.

3 Method

We first describe the domain generalization problem in Section 3.1. Then we move on to describe prefix tuning in Section 3.2. Finally, in Section 3.3, we describe our proposed approach, DAPA.

3.1 Problem Definition

Let $D^S = \{D^S_1, D^S_2, ..., D^S_n\}$ be the set of source domains. We denote the target domain with D^T. Domain generalization aims to seek a network which generalizes well on D^T when trained on D^S. We require our model to generate fluent summaries for target domain documents when trained on source domain documents.

3.2 Prefix Tuning

We utilize prefix tuning to train a separate prefix for each source domain. We begin by restating the

transformer attention:

$$\text{attn}(Q, K, V) = \text{softmax}(\frac{Q \cdot K^\top}{\sqrt{d}})V \quad (1)$$

Here, the query matrix Q, the key matrix K, and the value matrix V are obtained through independent linear transformations on the output of the previous layer/encoder. d is the model dimension. Note that we omit the multihead notation for clarity.

Prefix tuning modifies the transformer attention by adding tunable prefixes to K and V. Consequently K is modified as $K' = [h_K; K]$ and V is modified as $V' = [h_V; V]$. Here h_K and h_V represent the key prefix and the value prefix respectively.

Following Li and Liang (2021), we model these prefixes using a two layer MLP as follows:

$$\begin{aligned} h_K^j &= W_{K,2}^j f(W_{K,1}^j E^j + b_{K,1}^j) + b_{K,2}^j \\ h_V^j &= W_{V,2}^j f(W_{V,1}^j E^j + b_{V,1}^j) + b_{V,2}^j \end{aligned} \quad (2)$$

where $W_{K,1}^j, W_{V,1}^j, W_{K,2}^j, W_{V,2}^j \in \mathbb{R}^{d \times d}$ are trainable weights, and $b_{K,1}^j, b_{V,1}^j, b_{K,2}^j, b_{V,2}^j \in \mathbb{R}^d$ are trainable biases. $E^j \in \mathbb{R}^{C \times d}$ is a trainable embedding matrix with C as the prefix length. Index j corresponds to source domain D_j^S. We detail the initialization of E^j in Section 4.3. Each source prefix is trained in an end-to-end fashion on its corresponding source domain data.

3.3 Domain Aligned Prefix Averaging

Having formulated our problem and described prefix tuning, we now describe our approach, DAPA.

3.3.1 Computing Weights to Average Source Prefixes

DAPA utilizes the summary generation capabilities of source prefixes to generate weights for averaging source prefixes. Let $D_{m,sample}^T = \{x_1, x_2, ..., x_m\}$ be a set m unlabelled documents from the target domain. In our experiments, we observe that a value of m as small as 50 suffices. Let $P^S = \{P_1^S, P_2^S, ..., P_n^S\}$ represent the set of source prefixes. Note that $P_j^S = \{h_K^j, h_V^j\}$. For target domain document x_i, DAPA first generates n summaries pertaining to each source prefix as follows:

$$y_i^j = M(x_i; P_j^S);\ 1 \le i \le m,\ 1 \le j \le n \quad (3)$$

where M represents the frozen pretrained language model.

Next, it uses an encoder f to generate sentence representations for the summary y_i^j as $r_i^j = f(y_i^j)$ and the document x_i as $t_i = f(x_i)$. We use SentenceBERT (Reimers and Gurevych, 2019) as our encoder. Following this, we compute average document-summary cosine similarity scores for each source prefix as follows:

$$s^j = \sum_{i=1}^{m} \text{cosine-similarity}(r_i^j, t_i) \quad (4)$$

The final weights for averaging source prefixes are generated by taking a softmax over the average document-summary similarity scores as:

$$w^j = \frac{\exp(s^j)}{\sum_{i=1}^{n} \exp(s^i)} \quad (5)$$

3.3.2 Prefix Averaging

Given a target domain document x, we wish to generate a summary y with a target prefix obtained by averaging the source prefixes using our weight averaging scheme described in Section 3.3.1. Through $W = \{w^1, w^2, ..., w^n\}$, we take a weighted average of the source prefixes as follows:

$$\begin{aligned} h_K^T &= \sum_{j=1}^{n} w^j h_K^j \\ h_V^T &= \sum_{j=1}^{n} w^j h_V^j \\ P^T &= \{h_K^T, h_V^T\} \end{aligned} \quad (6)$$

where P^T is the target prefix through which the target summary $y = M(x; P^T)$ is generated. Note that test time averaging requires recomputation of h_K^j and h_V^j by replacing E^j with E^T in equation 2. We detail the computation of E^T in Section 4.3.

4 Experimental Setup

4.1 Dataset and Metrics

We use four summarization datasets, each belonging to a different domain. For the *news* domain we use the CNN/Daily Mail dataset (Hermann et al., 2015); Samsum (Gliwa et al., 2019) for the *chat* domain; Reddit posts for the *social-media* domain (Kim et al., 2018); ScisummNet (Yasunaga et al., 2019) for training on the *scientific* domain and Cl-SciSumm (Jaidka et al., 2018) for testing on the *scientific* domain. Dataset statistics are presented in Table 1. For evaluation, we report ROUGE-1, ROUGE-2 and ROUGE-L metrics[2] (Lin, 2004).

[2] https://github.com/pltrdy/files2rouge

Domain	Train size	Dev size	Test size
News	287,113	13,368	11,490
Scientific	947	10	10
Social-media	33246	4151	4155
Chat	14,732	818	819

Table 1: Dataset statistics for each domain.

4.2 Baselines

We use the method of empirical risk minimization (ERM) as our primary baseline. It trains the model by minimizing the sum of errors across source domains and examples. For computer vision tasks, Gulrajani and Lopez-Paz (2021) have shown that a well tuned ERM baseline performs competitively with several sophisticated methods for domain generalization. Thus, we use it as our primary baseline. We define two variants of ERM: i) ERM-finetune, finetunes the pretrained language model on a combination of all source domains. ii) ERM-prefix, prefix-tunes the backbone language model on a combination of all source domains. To validate the efficacy of our weight averaging scheme, we create two variants of DAPA, namely DAPA-average and DAPA-max. DAPA-average follows $w^j = \frac{1}{n}$ and DAPA-max takes a max pooling operation over the source prefixes instead of averaging them. We also present results for an instantaneous version of DAPA, DAPA-inst. Here, instead of using $D^T_{m,sample}$, we use the current testing document to compute weights W. We also consider four additional baselines as an upper bound to our method, results for which are presented in Appendix A.

4.3 Training Details

For our backbone pretrained model, we use T5-small (containing roughly 60M parameters) (Raffel et al., 2020). Prefix tuning adds rough 922K test time parameters to T5-small. Prefix length C is fixed to 50 unless otherwise specified. m is also set to 50. We verify our choices for C and m in Section 5. Both, finetuning and prefix tuning experiments are optimized with Adafactor (Shazeer and Stern, 2018). Finetuning uses a maximum learning rate of $5e{-}4$, a square root decay schedule, and a linear warmup of 5000 steps. Prefix tuning uses a constant learning rate of $5e-3$. All other Adafactor specific hyperparameters are left to their default values in HuggingFace-transformers[3] (Wolf et al., 2020). We utilize OpenPrompt[4] (Ding et al., 2022) and HuggingFace-transformers to implement prefix tuning, and use the sentence-transformers[5] implementation for SentenceBERT.

For finetuning, we employ a batch size of 5 with gradient accumulation up to 5 iterations. For prefix tuning, we use a batch size of 5 but without any gradient accumulation. All our experiments are run on a single Nvidia-RTX 2080 Ti machine. One finetuning weight update (via gradient accumulation) takes rough 224 milliseconds and one prefix tuning iteration takes roughly 139 milliseconds. All our models are trained for 10 epochs with early stopping performed through validation ROUGE scores. For ERM-finetune and ERM-prefix, the training process is stopped if the in-domain validation scores for any of the three source domains starts to fall. Each training experiment is carried out only once.

For prefix tuning, we initialize E^T with T5 embeddings of the C most frequent sentencepiece[6] (Kudo and Richardson, 2018) tokens of $D^T_{m,sample}$. For DAPA-inst, the same process is applied, however instead of $D^T_{m,sample}$, the current test document is used. For source domains, C most frequent tokens are extracted from the train set.

Summary generation uses a beam length of 10 and a repetition penalty of 2.5. All source documents are truncated to 512 sentencepiece tokens and all summaries are truncated to 200 sentencepiece tokens. For the *scientific* domain, we only include the document's abstract, introduction and conclusion in our input to present the document's most crucial aspects within T5's maximum allowed sequence length. All our results are presented on the test sets of the four domains.

4.4 Main Results

Table 2 presents results for our domain generalization experiments. DAPA outperforms all compared methods on the *chat* domain and outperforms all compared methods on two out of the three ROUGE scores on the *news* domain. Owing to our prefix averaging method, DAPA demonstrates better generalization capabilities when compared to ERM-finetune and ERM-prefix. DAPA-max and DAPA-average do not utilize the summary generation capabilities of source prefixes and, thus, fail to account

[3] https://github.com/huggingface/transformers

[4] https://github.com/thunlp/OpenPrompt

[5] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[6] https://github.com/google/sentencepiece

Approach	News			Scientific			Chat			Social-media		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
ERM-finetune	39.94	18.09	**32.90**	**33.72**	**22.14**	**31.66**	25.52	6.69	21.09	16.53	2.94	11.58
ERM-prefix	38.05	16.49	32.15	29.29	17.55	26.78	22.46	5.47	19.39	**17.5**	**3.01**	**12.36**
DAPA-average	35.98	15.90	30.58	28.78	17.04	26.52	23.04	5.55	19.68	16.86	2.88	11.88
DAPA-max	35.18	15.39	30.36	28.57	15.84	25.03	26.64	7.32	22.19	16.97	2.76	11.83
DAPA-inst	35.93	15.91	30.58	29.34	18.72	27.40	20.81	4.78	17.79	16.54	2.81	11.64
DAPA	**40.28**	**18.12**	32.78	30.84	18.97	27.23	**28.23**	**8.70**	**22.86**	14.48	2.68	9.70

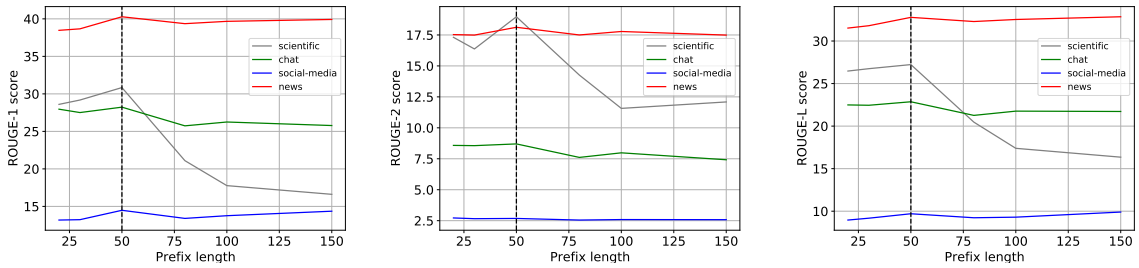
Table 2: ROUGE scores for domain generalization on the four domains.

Target Domain	Excluded Domain	R-1	R-2	R-L
Scientific	News	30.61	18.01	26.46
	Chat	30.20	19.54	26.78
	Social-media	31.25	19.26	27.26
Chat	News	28.23	8.67	22.86
	Social-media	28.23	8.67	22.86
	Scientific	25.63	6.51	21.16
Social-media	News	14.6	2.71	9.79
	Chat	14.6	2.71	9.79
	Scientific	17.25	3.04	12.24
News	Social-media	40.29	18.12	32.78
	Chat	40.27	18.12	32.78
	Scientific	38.06	16.71	32.28

Table 3: ROUGE scores while using only two source domains to compute W. The second column mentions the domain whose prefix has been left out for the target domain's prefix computation.

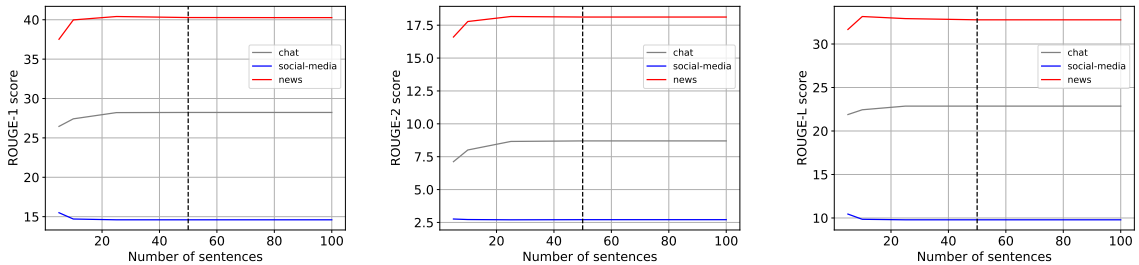
Target Domain	Source Domain	R-1	R-2	R-L
Scientific	News	36.73	25.01	34.07
	Chat	32.14	19.08	28.15
	Social-media	18.81	9.42	17.40
Chat	News	25.61	6.49	21.11
	Social-media	21.21	5.76	18.87
	Scientific	28.23	8.70	22.86
Social-media	News	17.22	3.01	12.23
	Chat	16.37	2.72	11.61
	Scientific	14.61	2.71	9.79
News	Social-media	30.47	13.49	27.14
	Chat	38.03	16.69	32.25
	Scientific	40.27	18.12	32.78

Table 4: ROUGE scores while using a single source domain's prefix to generate target domain summaries. The second column mentions the source domain adopted to generate summaries on the target domain.



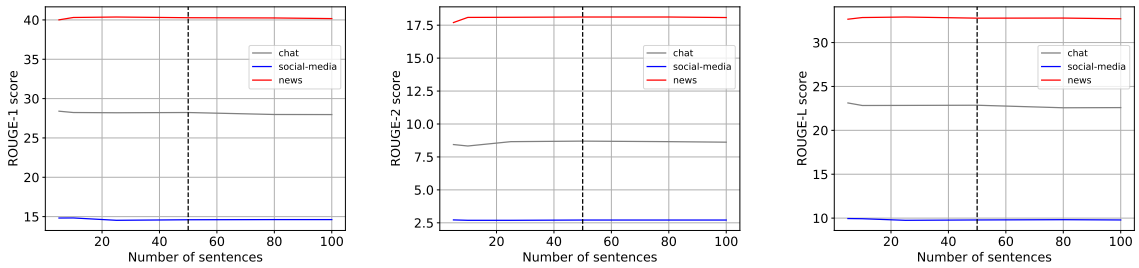
(a) ROUGE-1 variation. (b) ROUGE-2 variation. (c) ROUGE-L variation.

Figure 2: Variation of ROUGE scores with prefix length C. Empirically, $C = 50$ is the most optimal prefix length. Throughout this experiment, $m = 50$.



(a) ROUGE-1 variation. (b) ROUGE-2 variation. (c) ROUGE-L variation.

Figure 3: Variation of ROUGE scores with the number of sentences used for computing weights W. Throughout this experiment, $C = 50$ and E^T is initialized with 50 most frequent tokens from $D^T_{m,sample}$.



(a) ROUGE-1 variation. (b) ROUGE-2 variation. (c) ROUGE-L variation.

Figure 4: Variation of ROUGE scores with the number of sentences used for deriving the C most frequent words. Throughout this experiment, $C = 50$.

for the aspects most crucial to summary generation for the two domains. Also, DAPA-inst performs significantly worse than DAPA, thereby emphasizing the importance of using a greater number of target domain documents to better approximate the weights for averaging source prefixes.

On the contrary, ERM-finetune outperforms DAPA on the *scientific* domain. Yasunaga et al. (2019) demonstrate the superior performance of extractive approaches over abstractive summarization approaches for the *scientific* domain. Thus, the ability to copy phrases from the input document becomes imperative to a good performance. Possibly, since all model parameters are tuned for ERM-finetune, its ability to copy phrases from the input document exceeds that of DAPA which only tunes the source prefixes. Despite this, DAPA outperforms the other compared methods for reasons similar to the ones stated previously.

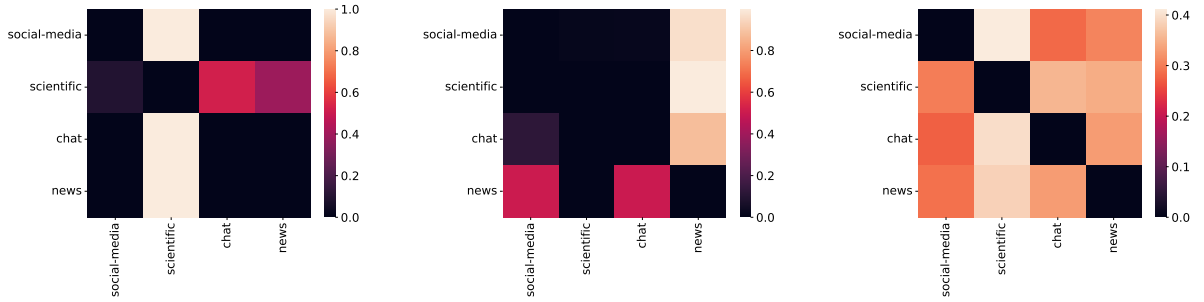
Also, all compared methods outperform DAPA on the *social-media* domain. We find that DAPA allocates a significant amount of weight to the *scientific* domain prefix. However, we observe that the *scientific* domain adversely affects performance on the *social-media* domain (Refer to Section 5.1 for more details). This irregularity may have resulted from the encoder f. Further investigation to this is left for future work. Only 27.94% of the maximal weights selected by DAPA-max belong to the *scientific* domain prefix. Also, DAPA-average assigns equal weights to all the three source domains which is less than the weight assigned by DAPA to the *scientific* domain. Thus both DAPA-average and DAPA-max outperform DAPA. The noise added by DAPA-inst to the weight calculation process results in a smaller weight assigned to the *scientific* domain prefix (0.35 vs 1.00) as a result of which DAPA-inst outperforms DAPA. Owing to the larger dataset size for the *chat* and *news* domains, ERM-prefix and ERM-finetune are less impacted by adverse effects of the *scientific* domain and thus outperform DAPA. ERM-finetune underperforms ERM-prefix probably because of its larger capacity to retain *scientific* domain knowledge.

5 Analysis

To study the impact of various factors in DAPA, and better understand its efficacy, we conduct a series of analysis.

5.1 Effect of Source Domains on the Target Domain

In Table 3, and Table 4, we analyze the effect of various source domains on the target domain. Throughout these experiments, $m = 50$ and $C = 50$. Note that the experiments in this section only require recomputation of the prefix averaging weights and thus support the claim that DAPA allows for computationally efficient addition of new source domains. For the *scientific* domain, we can see that the performance is best when only using the *news* domain, in fact, it outperforms ERM-finetune from Table 2. Adding the *chat* and *social-media* domains only hampers performance. The performance is worst when using only the *social-media* domain. The improvement over this result indicates that DAPA is able to assign appropriate weights to the three source domains allowing for a greater contribution from the *news* and *chat* domains. In real-world applications where the number of domains is significantly greater than our setting, and there is no labelled data to measure the performance over the target domain (to decide the optimal set source prefixes to be averaged), DAPA offers an effective scheme to choose the most appropriate source prefixes.



(a) Weights obtained by DAPA for averaging source prefixes.

(b) Probabilities assigned to the target domain by a BERT model trained for source domain identification.

(c) Probabilities assigned to the target domain by DAPA-alt.

Figure 5: Source domain preferences for the target domain obtained through DAPA, a BERT model trained for source domain identification and DAPA-alt.

In the case of the *chat* domain, ablating the *scientific* domain results in degraded performance. On the contrary, excluding the other two prefixes does not affect DAPA's performance. Similarly for the *news* domain, removing the *scientific* domain's prefix results in a significant drop in all three ROUGE scores and removing the *chat* domain results in a slight drop in ROUGE-1 score. Here, we see that both the *chat*, and the *scientific* domain contribute to the performance since using only one of them underperforms the result in Table 2.

In the case of the *social-media* domain, excluding the *scientific* domain significantly improves DAPA's performance. Also, both the *news* and the *chat* domain contribute to the performance since using only one of them underperforms their weighted average.

5.2 Effect of Prefix Length C

An analysis on the effect of C is presented in Figure 2. C most frequent words in the target domain are extracted from $m = 50$ documents. In general, we observe the target domain performance increases up to $C = 50$ following which it either drops or remains more or less the same. Thus, we select $C = 50$ for DAPA. $scientific$ domain ROUGE scores drop significantly after 50 prefix tokens. We leave further exploration into this for future work.

5.3 Effect of m on W

An analysis on the effect of the number of sentences used for computing weights to average source prefixes is presented in Figure 3. Beyond $m = 20$, the performance on the target domain remains more or less constant. Thus, we stick to our initial choice of using 50 sentences to compute weights W.

An analysis on the effect of the number of sentences used for obtaining C most frequent tokens is presented in Figure 4. Again, the performance does not vary significantly beyond $m = 20$. Thus, we hold to our initial choice of $m = 50$ for our main experiments. Note that we do not include results on the *scientific domain* for this subsection since its test set has only 10 instances, and we use all of them for our experiments.

5.4 Does Averaging over Source E^js Help?

In our main method, we initialize E^T with C most frequent sentencepiece tokens from the m unlabelled documents. Here, we explore an alternative way of initializing E^T wherein we use w^js to take a weighted average of source E^js :

$$E^T = \sum_{j=1}^{n} w^j E^j \tag{7}$$

Results for this initialization scheme (DAPA-embed) are presented in Table 5. DAPA outperforms DAPA-embed across domains demonstrating the benefits of supplying P^T with some prior target domain knowledge by initializing E^T with C most frequent sentencepiece tokens from the m unlabelled documents.

5.5 Does Softmax before Summation Help?

Here, we propose an alternative way of computing w^j. Instead of summing over the document-summary cosine similarities (Equation 4) and then applying the softmax operation (Equation 5), we first apply the softmax operation to document-summary similarity scores following which we average over them. That is, we replace Equation 4

Approach	News			Scientific			Chat			Social-media		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
DAPA-embed	39.06	17.96	31.81	28.92	16.79	26.24	26.32	8.21	21.31	13.04	**2.76**	8.80
DAPA	**40.28**	**18.12**	**32.78**	**30.84**	**18.97**	**27.23**	**28.23**	**8.70**	**22.86**	**14.48**	2.68	**9.70**

Table 5: ROUGE scores for initializing E^T as a weighted average of E^js, i.e. DAPA-embed.

Approach	News			Scientific			Chat			Social-media		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
DAPA-alt	36.12	15.96	30.70	29.42	18.02	27.32	23.57	5.84	20.05	**16.81**	**2.90**	**11.82**
DAPA	**40.28**	**18.12**	**32.78**	**30.84**	**18.97**	**27.23**	**28.23**	**8.70**	**22.86**	14.48	2.68	9.70

Table 6: ROUGE scores for an alternative way of computing w^js as discussed in Section 5.5, i.e. DAPA-alt.

with:

$$w_i^j = \frac{\exp(\text{cosine-similarity}(r_i^j, t_i))}{\sum_{k=1}^{n} \exp(\text{cosine-similarity}(r_i^k, t_i))} \quad (8)$$

and Equation 5 with

$$w^j = \frac{1}{m} \sum_{i=1}^{m} w_i^j \quad (9)$$

By doing so, we are flattening the weights w^j. This is evident from Figure 5c where the target domain assigns near equal weights to all three source domains. This is different from DAPA, where the weights w^j are sharp as depicted in Figure 5a. In Table 6, DAPA outperforms this alternative strategy (DAPA-alt) on three out of the four domains. A flattened w^j distribution approaches DAPA-average, and thus, does not benefit from DAPA's weight averaging scheme. DAPA-alt outperforms DAPA on the *social-media* domain since it assigns near equal weights to each source domain (Refer to Section 4.4 for a detailed analysis).

5.6 Is DAPA correlated to Document Similarity?

We analyze how DAPA's weight assigning process aligns with source-target domain similarity at the document level. For this we train BERT-base (Devlin et al., 2019) on 300 source domain documents (100 from each source domain) for source domain identification. We evaluate this model on the target domain's $D^T_{50,sample}$. We plot the average probabilities assigned to each source domain in Figure 5b. Also, in Figure 5a, we plot weights W computed by DAPA. Note that BERT-base achieves perfect test accuracy when evaluated on an in-domain validation set for each training setting. Entries of the form $[x, x]$ are always zero owing to our domain generalization setting.

From the two plots, it is clear that DAPA's weight assignment process does not always correlate with source-target domain similarity at the document level. For the *news* domain, as per Table 3 and Table 4, the *scientific* domain contributes most to the model's performance, on the other hand, BERT assigns near equal probability to the *social-media* and *chat* domain, and assigns near zero probability to the *scientific* domain. Similarly, for the *chat* domain, the *scientific* domain is vital to good performance, however, BERT assigns a low probability to it. Whereas, for the *social-media* and *scientific* domain, BERT does a better job and assigns a higher probability to the *news* domain. These results navigate us to the conclusion that DAPA does not use document level similarities and indeed relies on the summary generation capabilities of source prefixes.

6 Conclusion

In this paper, we present DAPA, a lightweight, domain aligned prefix averaging approach to domain generalization in abstractive summarization. DAPA utilizes source prefixes to generate summaries for a small number of target domain documents. The similarity of these summaries to their corresponding documents are used for calculating weights required to average source prefixes. DAPA can easily account for the addition of new source domains since only the prefix averaging weights need to be recomputed. On four diverse summarization domains, DAPA either performs comparably or outperforms the baselines. We also perform an in-depth analysis of various components of DAPA to further strengthen our design choices. In future, we would like to develop an improved similarity function f and analyze the loss landscapes of these models to corroborate our prefix averaging strategy.

7 Limitations

Our work focuses on domain generalization for abstractive summarization through prefix averaging. However, we do not experiment with larger backbone models due to computational constraints. Based on previous works we expect our approach's performance to improve with model size. Also, a larger sequence length for prefix tuning increases the computational costs at inference.

Another limitation of our work is that we do not test it on natural language understanding tasks. This can be part of a future work.

8 Ethical Statement

We consider our approach to have low ethical risks since we do not utilize any data biases. Our approach could be extended to any natural language generation task and does not constraint the input/output structure. We therefore conclude that our method would not bring any harmful ethical impact.

References

Devansh Arpit, Huan Wang, Yingbo Zhou, and Caiming Xiong. 2021. Ensemble of averages: Improving model selection and boosting performance in domain generalization. *CoRR*, abs/2110.10832.

Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. 2021. SWAD: domain generalization by seeking flat minima. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 22405–22418.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Haitao Zheng, and Maosong Sun. 2022. Openprompt: An open-source framework for prompt-learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL 2022 - System Demonstrations, Dublin, Ireland, May 22-27, 2022*, pages 105–113. Association for Computational Linguistics.

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, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 4830–4842. Association for Computational Linguistics.

Xiangyu Duan, Hongfei Yu, Mingming Yin, Min Zhang, Weihua Luo, and Yue Zhang. 2019. Contrastive attention mechanism for abstractive sentence summarization. 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)*. Association for Computational Linguistics.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P. Vetrov, and Andrew Gordon Wilson. 2018. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8803–8812.

Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. 2018. Bottom-up abstractive summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4098–4109. Association for Computational Linguistics.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. *ArXiv*, abs/1911.12237.

Ishaan Gulrajani and David Lopez-Paz. 2020. In search of lost domain generalization. *ArXiv*, abs/2007.01434.

Ishaan Gulrajani and David Lopez-Paz. 2021. In search of lost domain generalization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Kokil Jaidka, Michihiro Yasunaga, Muthu Kumar Chandrasekaran, Dragomir R. Radev, and Min-Yen Kan. 2018. The cl-scisumm shared task 2018: Results and key insights. In *Proceedings of the 3rd Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2018) co-located with the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*

(SIGIR 2018), Ann Arbor, USA, July 12, 2018, volume 2132 of *CEUR Workshop Proceedings*, pages 74–83. CEUR-WS.org.

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2018. Abstractive summarization of reddit posts with multi-level memory networks. *CoRR*, abs/1811.00783.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pages 66–71. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex Chichung Kot. 2018. Domain generalization with adversarial feature learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5400–5409.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Annual Meeting of the Association for Computational Linguistics*.

Yixin Liu and Pengfei Liu. 2021. Simcls: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 1065–1072. Association for Computational Linguistics.

Michael Matena and Colin Raffel. 2021. Merging models with fisher-weighted averaging. *CoRR*, abs/2111.09832.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, 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, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, pages 280–290. ACL.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Chengwei Qin and Shafiq Joty. 2021. LFPT5: A unified framework for lifelong few-shot language learning based on prompt tuning of T5. *CoRR*, abs/2110.07298.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Alexandre Ramé, Matthieu Kirchmeyer, Thibaud Rahier, Alain Rakotomamonjy, Patrick Gallinari, and Matthieu Cord. 2022. Diverse weight averaging for out-of-distribution generalization. *ArXiv*, abs/2205.09739.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *ArXiv*, abs/1908.10084.

Timo Schick and Hinrich Schütze. 2020. Few-shot text generation with pattern-exploiting training. *CoRR*, abs/2012.11926.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. *CoRR*, abs/1704.04368.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4603–4611. PMLR.

Zhixing Tan, Xiangwen Zhang, Shuo Wang, and Yang Liu. 2021. MSP: multi-stage prompting for making pre-trained language models better translators. *CoRR*, abs/2110.06609.

Thuy-Trang Vu, Shahram Khadivi, Dinh Q. Phung, and Gholamreza Haffari. 2022. Domain generalisation of NMT: fusing adapters with leave-one-domain-out training. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 582–588. Association for Computational Linguistics.

Bailin Wang, Mirella Lapata, and Ivan Titov. 2020a. Meta-learning for domain generalization in semantic parsing. *ArXiv*, abs/2010.11988.

Yufei Wang, Haoliang Li, and Alex C. Kot. 2020b. Heterogeneous domain generalization via domain mixup.

In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 3622–3626. IEEE.

Zhen Wang. 2020. Unseen target stance detection with adversarial domain generalization. *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Michihiro Yasunaga, Jungo Kasai, Rui Zhang, A. R. Fabbri, Irene Li, Dan Friedman, and Dragomir R. Radev. 2019. Scisummnet: A large annotated corpus and content-impact models for scientific paper summarization with citation networks. In *AAAI*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Lulu Zhao, Fujia Zheng, Keqing He, Weihao Zeng, Yuejie Lei, Huixing Jiang, Wei Wu, Weiran Xu, Jun Guo, and Fanyu Meng. 2021. Todsum: Task-oriented dialogue summarization with state tracking. *CoRR*, abs/2110.12680.

Lulu Zhao, Fujia Zheng, Weihao Zeng, Keqing He, Weiran Xu, Huixing Jiang, Wei Wu, and Yanan Wu. 2022. Domain-oriented prefix-tuning: Towards efficient and generalizable fine-tuning for zero-shot dialogue summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 4848–4862. Association for Computational Linguistics.

A Additional Results

For the sake of completeness, we present four additional baselines in Table 7. Finetune-target assumes that the m target domain documents are labelled and thus uses standard finetuning to update T5-small parameters with these m documents. Likewise, Prefix-target uses prefix tuning to train a prefix from scratch with these m documents. Since, the above two baselines use labelled documents they cannot be directly compared with DAPA and only present an upper bound to the performance of DAPA. Using m labelled documents reaps benefits on three out of the four domains. Note that we use $m = 50$ for the above two baselines.

We also present results for full finetuning (Finetune) and full prefix tuning (Prefix) on each of the four domains. Finetune uses the target domain's entire training set to update T5-small parameters and Prefix trains a prefix from scratch on the target domain's training set. Again, these two baselines only act us upper bounds to DAPA and outperform it across domains.

Approach	News			Scientific			Chat			Social-media		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Finetune-target	37.65	17.05	32.06	39.66	21.76	35.57	32.65	12.43	29.03	17.38	2.92	12.77
Prefix-target	38.10	17.61	33.11	49.88	30.64	43.19	39.30	15.67	33.49	19.48	4.10	15.35
Finetune	41.20	19.18	35.20	52.97	32.27	45.85	49.42	24.31	42.43	25.06	7.23	19.96
Prefix	40.56	19.07	34.94	49.58	29.72	42.84	46.20	22.48	40.27	23.22	6.30	18.59
DAPA	40.28	18.12	32.78	30.84	18.97	27.23	28.23	8.70	22.86	14.48	2.68	9.70

Table 7: Results for the four additional baselines.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 7

☑ A2. Did you discuss any potential risks of your work?
Section 8

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4.1, Section 4.2 and Section 4.3

☑ B1. Did you cite the creators of artifacts you used?
Section 4.1, Section 4.2 and Section 4.3

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Since all of them are made publicly available on the internet for researchers to use.

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Since we only present our work from the research point of view and do not plan on deploying it.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Our work following previous works on domain generalization does not attempt this.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 4.1 Section 4.2 and Section 4.3

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Table 1

C ☑ Did you run computational experiments?

Section 4.1, Section 4.2 Section 4.3, Section 5.1, Section 5.2, Section 5.3 and Section 5.4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 4.3

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 4.1, Section 4.2 Section 4.3, Section 5.1, Section 5.2, Section 5.3 and Section 5.4

☒ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Our algorithm comes into play only after the training phase is over. Thus all parameters are frozen. All our models are trained only once which we have mentioned in the paper.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 4.1, Section 4.2 and Section 4.3

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.