Rethinking Domain Generalization for Text: Classifier-Only Training with Test-Time Domain Shift Correction

Anonymous ACL submission

Abstract

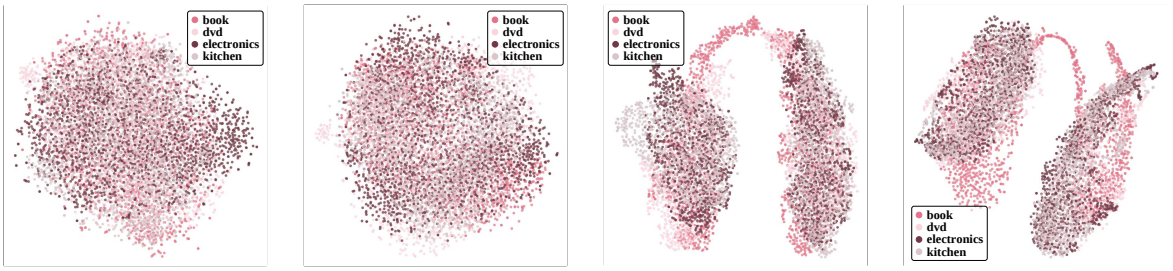
Domain generalization aims to enable models trained on source domains to generalize to previously unseen target domains. In text classification, most existing methods explicitly learn domain-invariant representations through adversarial or contrastive objectives on top of pretrained language models (PLMs). However, PLMs are already trained on massive corpora and inherently encode substantial domain-invariant semantics, making additional representation learning potentially unnecessary and even harmful due to source-domain overfitting. In this work, we rethink text domain generalization from a representation-preserving perspective and propose **Classifier Only for Source** (CO4S), a simple plug-in framework. CO4S fully freezes the pretrained backbone and trains only a lightweight classifier on the source domain(s). At test time, it estimates a domain shift vector from the difference between the mean sentence representations of the source and target domains, and applies this shift to align target representations before classification. This enables effective test-time domain shift correction without modifying the backbone or introducing complex training objectives. Experiments on six benchmark datasets show that CO4S consistently improves generalization performance across multiple evaluation protocols, achieving an average macro-F1 gain of 5.51% with as few as 7.83 KB of trainable parameters. These results suggest that effective text domain generalization can be achieved by better exploiting the intrinsic invariance of pretrained language models.

1 Introduction

Pretrained language models (PLMs) have become the dominant paradigm for a wide range of natural language processing (NLP) tasks. In practice, these models are often fine-tuned on task-specific datasets to achieve strong in-domain performance. However, such fine-tuning can easily lead to overfitting to the source data distribution, consequently impairing generalization to unseen domains (Guo and Yu, 2022). To address this challenge, domain generalization (DG) aims to train models on one or multiple source domains such that they can generalize effectively to previously unseen target domains (Wang et al., 2023a).

Domain generalization has been extensively studied in computer vision, where domain shifts (e.g., between paintings, photographs, and sketches) are often visually apparent and structurally distinct (Wang et al., 2023b). In contrast, domain gaps in natural language are more subtle and less explicit, even within the same language (Ling et al., 2024). For example, PLMs such as GPT (Ouyang et al., 2022), Qwen (Yang et al., 2024), and DeepSeek (DeepSeek-AI et al., 2024) are typically trained on large-scale public corpora and cannot directly access confidential or domain-restricted data (e.g., government documents or clinical records). As a result, models that perform well on general-domain data may still struggle to generalize to such unseen domains without additional adaptation.

To improve text domain generalization, existing methods predominantly focus on *learning domain-invariant representations*. Representative approaches can be broadly categorized into two paradigms. (1) **Adversarial training-based methods** employ a min–max objective to encourage representations that are indistinguishable across domains by fooling a domain discriminator (Jia and Zhang, 2022, 2023; Bhattacharjee et al., 2024; Chen et al., 2025). (2) **Contrastive learning-based methods** leverage instance-level or prototype-level contrastive objectives to align representations across multiple source domains, thereby promoting domain invariance in the embedding space (Tan et al., 2022; Bhattacharjee et al., 2024). Despite their methodological differences, these approaches share a common assumption: *explicitly*



(a) Sentence representation distributions without fine-tuning. (b) Sentence representation distributions after PDA-based fine-tuning. (c) Sentence representation distributions after EAGLE-based fine-tuning. (d) Sentence representation distributions after MSCL-based fine-tuning.

Figure 1: T-SNE (van der Maaten and Hinton, 2008) visualization of sentence representations from the RoBERTa-base-encoded (Liu et al., 2019) Amazon dataset, where "book" is the target domain. (a) Untuned representations. (b-d) Representations fine-tuned with PDA (Jia and Zhang, 2022), EAGLE (Bhattacharjee et al., 2024), and MSCL (Tan et al., 2022), respectively.

enforcing domain invariance through additional training objectives is necessary for effective generalization.

In this work, we revisit this assumption from the perspective of pretrained language models. We argue that this prevailing view overlooks a critical yet underexplored fact: **PLMs, by virtue of large-scale pretraining, already encode substantial domain-invariant semantic representations**. As illustrated in Fig. 1(a), sentence embeddings extracted from an untuned RoBERTa-base model already exhibit a high degree of cross-domain overlap on the Amazon Reviews dataset (Blitzer et al., 2007). In contrast, Fig. 1(b)–Fig. 1(d) qualitatively show that applying different domain generalization objectives can alter the geometric structure of the representation space, sometimes resulting in less overlapping embeddings across domains. While such visualizations do not directly quantify representation quality, they motivate us to reconsider whether explicitly enforcing additional domain invariance always preserves the intrinsic cross-domain semantics learned during pretraining.

At first glance, this perspective may appear related to prior feature alignment or moment-matching approaches that aim to reduce distributional discrepancies between source and target domains. However, a key distinction lies in *where and how* alignment is performed. Existing methods typically enforce alignment by modifying the representation function through additional training objectives, thereby reshaping the embedding space itself (Jin et al., 2020; Nguyen et al., 2022; Wei et al., 2025). In contrast, our goal is not to *learn* aligned representations, but to *preserve* the pretrained representation geometry of PLMs and instead adjust the relative position between a fixed source-trained classifier and target representations at inference time.

Motivated by this insight, we propose to *rethink text domain generalization* from a lightweight and representation-preserving perspective. Instead of re-optimizing the backbone encoder to learn additional invariances, we ask a simpler question: *can we better exploit the domain-invariant representations that PLMs already provide?* To this end, we introduce **C**lassifier **O**nly for **S**ource (CO4S), a simple yet effective plug-in framework for text domain generalization. CO4S freezes the pretrained backbone entirely and trains only a lightweight classifier on the source domain(s). At test time, it estimates a domain shift vector by computing the difference between the mean sentence representations of the source and target domains, and applies this shift to align target-domain representations before classification.

Importantly, this mean shift operation should be interpreted as a lightweight test-time representation translation rather than a learned feature alignment mechanism. CO4S does not alter the representation function or enforce distributional matching during training. Instead, it performs a deterministic geometric correction that leverages the fixed decision boundary induced by the source-trained classifier.

This design yields a highly parameter-efficient and inference-time adaptation strategy. Without modifying the backbone or introducing complex training objectives, CO4S achieves consistent improvements across diverse benchmarks. In extensive experiments covering six datasets and multiple

evaluation protocols, CO4S improves macro-F1 by an average of 5.51% while introducing as few as 7.83 KB of trainable parameters.

The main contributions of this paper are summarized as follows:

1) We rethink text domain generalization by highlighting the overlooked role of intrinsic domain-invariant representations in pretrained language models, and show that explicitly learning additional invariances is not always necessary.

2) We propose CO4S, a classifier-only framework that performs test-time domain shift correction while fully preserving the pretrained backbone, offering a simple and effective alternative to existing DG methods.

3) We provide theoretical analysis showing that CO4S reduces the divergence between source and target domains and tightens the upper bound on target risk.

4) Extensive experiments demonstrate that CO4S achieves strong and consistent generalization performance with minimal parameter and computational overhead, making it practical for real-world deployment.

2 Methodology

Problem Formulation. The whole dataset $\mathcal{D}$ is divided into $N(N \geq 2)$ parts. Each part consists of texts and labels, $\mathcal{D}_i = \{\boldsymbol{\mathcal{T}}_i, \boldsymbol{\mathcal{Y}}_i\}$, where $i \in [1, N]$, $\boldsymbol{\mathcal{T}}_i$ and $\boldsymbol{\mathcal{Y}}_i$ are the text and label sets in the i-th domain, respectively. In DG settings, text distributions exhibit significant variation across different domains, $\mathcal{P}(\boldsymbol{\mathcal{T}}_i) \neq \mathcal{P}(\boldsymbol{\mathcal{T}}_j)(i \neq j)$ (where $\mathcal{P}(\boldsymbol{\mathcal{T}}_i)$ denotes the text distribution of the i-th domain), but the labels across all domains share a common space, $\boldsymbol{\mathcal{Y}}_i = \boldsymbol{\mathcal{Y}}_j$. During the training phase, all domains are partitioned into source and target domains ($\mathcal{D}^{src}$ and $\mathcal{D}^{tar}$), where the model can only access the texts and labels in $\mathcal{D}^{src}$ while remaining completely blind to the texts and labels in $\mathcal{D}^{tar}$.

Overview. As shown in Figure 2, the proposed method consists of three main steps: (1) training a source-domain-only classifier, (2) computing the domain shift vector via mean representation subtraction, and (3) applying the shift vector to target representations during inference.

Training the Classifier on Source Domain. As shown in Figs. 1(c) and 1(d), jointly learning the backbone and classifier during the training stage can lead to overfitting to the source domain, which undermines the domain-invariant features inherently learned by PLMs. To address this issue, we opt to train only a source-domain-specific classifier during this phase. This strategy preserves the pretrained model's domain-general knowledge while still enabling effective performance on downstream tasks. Given labeled text samples $\boldsymbol{\mathcal{T}}^{src}$ from N^{src} source domains, we encode them using a pretrained backbone f whose parameters θ_f are fixed:

$$\boldsymbol{h}^{src} = f(\boldsymbol{\mathcal{T}}^{src}; \theta_f), \tag{1}$$

where $\mathbf{h}_{\text{src}}$ denotes the sentence embeddings. These are passed to a trainable classifier f_c with parameters θ_c to obtain predictions $\tilde{\boldsymbol{\mathcal{Y}}}^{src} = \sigma(f_c(\boldsymbol{h}^{src}; \theta_c))$, where $\sigma(\cdot)$ denotes the softmax activation. The classifier is trained using cross-entropy loss:

$$\mathcal{L}_{\text{ce}} = -\sum_i \boldsymbol{\mathcal{Y}}_i^{src} \log \tilde{\boldsymbol{\mathcal{Y}}}_i^{src}. \tag{2}$$

This step enables the model to obtain a well-trained source domain classifier f_c^t, which serves as a lightweight "adapter" for the downstream tasks. Importantly, this process does not interfere with the internal knowledge encoded in the backbone.

Domain Shift Vector Computation. The goal of this stage is to capture the overall feature representation on each domain and to obtain the offset between the source and target domains. Specifically, during inference, we use Equation 1 to extract sentence representations for both source and target domains ($\boldsymbol{h}^{src}$ and $\boldsymbol{h}^{tar}$), respectively. Then we compute the centroids of all sentence representations in each domain to characterize the global distribution of the domain, as the following formula shows:

$$\boldsymbol{c}^{src} = \frac{1}{|\mathcal{D}^{src}|} \sum_{i=1}^{|\mathcal{D}^{src}|} \boldsymbol{h}_i^{src}, \boldsymbol{c}^{tar} = \frac{1}{|\mathcal{D}^{tar}|} \sum_{i=1}^{|\mathcal{D}^{tar}|} \boldsymbol{h}_i^{tar}. \tag{3}$$

Previous works have demonstrated the word analogy phenomenon (Mikolov et al., 2013; Ethayarajh et al., 2019), exemplified by the classic case of $\vec{\text{King}} - \vec{\text{Man}} + \vec{\text{Woman}} \approx \vec{\text{Queen}}$. Subsequent researches have also revealed that both the internal structures of pretrained language models (Ethayarajh, 2019) and their contextual representations (Reif et al., 2019) exhibit linearity. These findings support the hypothesis that semantic shifts (such as those between domains) can be

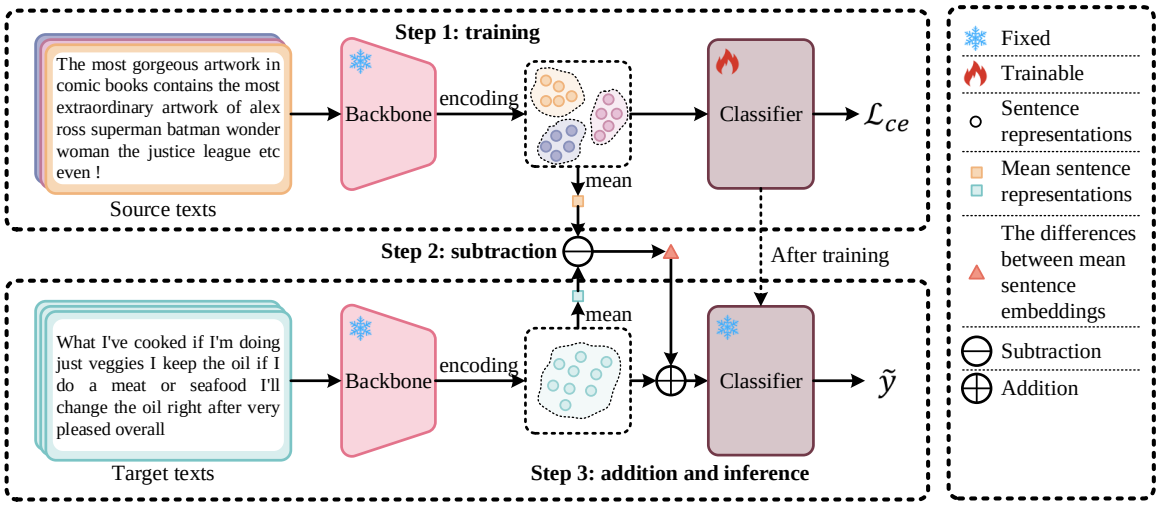


Figure 2: The workflow of CO4S.

approximated via linear translations in embedding space. Our method leverages this linearity by applying a mean shift vector between domains to compensate for domain shift. Here, the domain shift vector is defined as $\boldsymbol{d} = \boldsymbol{c}^{src} - \boldsymbol{c}^{tar}$. This vector serves as a coarse alignment offset[1], which we assume is sufficient to adjust the target embeddings closer to the source distribution in the embedding space.

Inference with Representation Adjustment. For each target sentence embedding $\boldsymbol{h}_i^{tar}$, we apply the domain shift vector $\boldsymbol{d}$ and classify using the previously trained classifier $\tilde{\boldsymbol{y}}^{tar} = \sigma(f_c^t(\boldsymbol{h}_i^{tar} + \boldsymbol{d}; \theta_c))$.

This final step effectively projects target domain representations into a region of feature space better aligned with the source domain, without altering the model's parameters.

Summary. Compared to existing approaches, our method trains only a single classifier on the source domain(s), making it agnostic to the number of source domains. Consequently, CO4S is equally applicable to both single-source and multi-source domain generalization settings. Moreover, our domain shift adjustment is performed entirely at inference time. Unlike prevailing feature alignment methods, CO4S does not require: 1) additional, complex learning strategies to explicitly learn domain-invariant representations; or 2) any modification to the hidden states output by the pre-trained language model. This design renders CO4S both lightweight and highly compatible with frozen PLMs, while maintaining strong generalization performance.

3 Experiments

3.1 Evaluation Setups

3.1.1 Datasets

We conduct experiments on two text classification tasks: sentiment analysis (SA) and natural language inference (NLI). Each task involves three datasets. Specifically, SA includes Amazon Reviews (Blitzer et al., 2007)[2], SST-2 (Socher et al., 2013), and IMDB (Maas et al., 2011), while NLI covers MNLI (Wang et al., 2019), SNLI (Kochkina et al., 2018a), and SICK (Marelli et al., 2014). The Amazon Reviews dataset consists of four domains, namely book (B), dvd (D), electronics (E), and kitchen (K). Meanwhile, the MNLI dataset involves five domains, which are fiction (F), government (G), slate (S), telephone (T), and travel (T*). Please refer to Appendix B.1 for the detailed statistical analysis of these datasets.

3.1.2 Implementation Details

We adopt RoBERTa-base (Liu et al., 2019)[3], DistilRoberta-base (Sanh et al., 2019)[4], BERT (De-

[1]Since in large language models (LLMs), sentence embeddings are often represented by word embeddings, our method can also be regarded as a form of word analogy (for example, encoder-only models typically use the embedding of the [CLS] token, while decoder-only models often use the embedding of the [EOS] token).

[2]https://github.com/jiachenwestlake/PDA

[3]https://huggingface.co/FacebookAI/roberta-base

[4]https://huggingface.co/distilbert/distilroberta-base

Table 1: The LODO results on the Amazon dataset with Roberta-base as backbone (%).

Model	NTP	BDE→K	BDK→E	BEK→D	DEK→B	Avg.
PDA	2.4165	70.25 ± 24.62	68.12 ± 26.72	69.72 ± 21.07	70.55 ± 20.94	69.66 ± 0.94
EAGLE	2.5969	85.01 ± 2.96	82.90 ± 1.92	81.63 ± 1.01	79.59 ± 2.30	82.28 ± 1.97
MSCL	0.7887	71.98 ± 1.44	66.71 ± 2.03	74.49 ± 1.13	74.62 ± 1.36	71.95 ± 3.20
SwAV	1.2485	65.51 ± 4.77	62.64 ± 4.17	58.12 ± 8.71	63.16 ± 1.12	62.36 ± 2.67
ELS	1.1858	84.51 ± 0.16	85.62 ± 0.40	83.85 ± 0.21	81.82 ± 1.97	83.95 ± 1.38
DomCLP	1.1172	88.18 ± 0.25	85.13 ± 0.32	82.71 ± 0.06	84.06 ± 0.28	85.02 ± 2.01
Backbone	**0.0015**	85.47 ± 0.69	85.25 ± 0.17	83.75 ± 0.13	80.79 ± 1.92	83.82 ± 1.87
CO4S	**0.0015**	**88.90 ± 0.19**	**85.73 ± 0.13**	**84.08 ± 0.06**	**84.59 ± 0.43**	**85.83 ± 1.87**
Improve	↓ 99.81%	↑ 0.72	↑ 0.11	↑ 0.23	↑ 2.77	↑ 1.87

Table 2: The LODO results on the MNLI dataset with Roberta-base as backbone (%).

Model	NTP	FGST→T*	FGST*→T	FGTT*→G	FSTT*→G	GSTT*→F	Avg.
PDA	4.1906	EG	EG	EG	EG	EG	-
EAGLE	4.3718	38.59 ± 6.10	39.15 ± 3.36	45.38 ± 0.52	39.56 ± 2.32	46.66 ± 3.39	41.87 ± 3.43
MSCL	0.7893	31.81 ± 1.77	34.25 ± 0.28	32.89 ± 0.84	33.92 ± 1.16	34.79 ± 2.55	33.53 ± 1.06
SwAV	1.2493	20.66 ± 2.96	18.77 ± 2.49	23.31 ± 0.45	19.94 ± 14.11	14.97 ± 11.45	19.53 ± 2.73
ELS	2.3701	**54.65 ± 0.31**	45.34 ± 2.68	48.33 ± 1.61	51.81 ± 1.47	49.53 ± 2.76	49.93 ± 3.15
DomCLP	1.1180	35.33 ± 0.87	39.06 ± 3.25	39.82 ± 3.37	37.42 ± 2.48	38.85 ± 2.65	38.09 ± 1.59
Backbone	**0.0023**	51.32 ± 1.83	44.45 ± 3.66	51.00 ± 0.94	51.93 ± 3.53	46.16 ± 3.49	48.97 ± 3.06
CO4S	**0.0023**	54.52 ± 0.85	**47.69 ± 1.44**	**51.93 ± 0.43**	**55.23 ± 0.88**	**50.29 ± 2.13**	**51.93 ± 2.77**
Improve	↓ 99.71%	↓ 0.13	↑ 2.35	↑ 0.93	↑ 3.30	↑ 0.76	↑ 2.00

vlin et al., 2019)[5] and Qwen2.5 (Yang et al., 2024) as the backbone model. For MLM models, we train for 15 epochs with a batch size of 16, setting the learning rate to 2e-2 and weight decay to 1e-2. The maximum sequence length is set to 128 for SA tasks and 48 per sentence for NLI tasks. For the 0.5B[6] and 1.5B[7] Qwen2.5 models, we train for 10 epochs with a batch size of 8, a learning rate of 5e-4, and a weight decay of 1e-2. For the 7B Qwen2.5 model[8], we train for 30 epochs with a batch size of 8, a learning rate of 1e-5, and a weight decay of 1e-2. The maximum sequence lengths are kept the same as those used for the MLM models. We train the MLM models, Qwen2.5-0.5B-Instruct, and Qwen2.5-1.5B-Instruct models on a single V100 GPU, and the Qwen2.5-7B-Instruct model on 8 H20 GPUs. All models are optimized using the AdamW (Loshchilov and Hutter, 2019) optimizer. All methods are run three times. Unless otherwise specified, we report the mean and standard deviation of the three runs throughout the paper.

We adopt macro-F1 as the evaluation metric. In all experiments, bold denotes the highest score and underlined indicates the second-highest. Unless otherwise specified, "improve" refers to the absolute gain in F1 over the second-highest score, and the percentage reduction in the number of trainable parameters (NTP) compared to the second-smallest model (M). Additionally, since PDA adopts adversarial training strategies, the optimization process tends to encounter numerous local saddle points. This makes the models prone to gradient explosion and unable to produce valid classification outputs. We denote such cases as EG (Exploding Gradients).

We adopt the Leave-One-Domain-Out (LODO) experimental setting. In this setting, we adopt $N-1$ domains as the source domain and test the model on 1 target domain, $\mathcal{D}^{src} \in \mathcal{D}$, $\mathcal{D}^{tar} \in \mathcal{D}$, $\mathcal{D}^{src} \cap \mathcal{D}^{tar} = \emptyset$, $|\mathcal{D}^{src}| = N-1$, and $|\mathcal{D}^{tar}| = 1$.

3.2 Baseline Methods

For MLM-based models, we compare against the backbone, PDA (Jia and Zhang, 2022) (an adversarial training method), EAGLE (Bhattacharjee et al., 2024) (a data augmentation-based contrastive learning method), MSCL (Tan et al., 2022) (a contrastive learning method), SwAV (Caron et al., 2020) (a representation learning method with data augmentation), ELS (Zhang et al., 2023) (environment label smoothing-based adversarial training method), and DomCLP (Lee et al., 2025) (a method based on prototype mixup and contrastive learning). For Qwen-based models, we only compare against the backbone. In all methods, the backbone parameters are frozen, while the parameters of other modules

[5]https://huggingface.co/google-bert/bert-base-uncased
[6]https://huggingface.co/Qwen/Qwen2.5-0.5B-Instruct
[7]https://huggingface.co/Qwen/Qwen2.5-1.5B-Instruct
[8]https://huggingface.co/Qwen/Qwen2.5-7B-Instruct

(e.g., the adversarial training module in PDA) remain learnable. For simplicity, we denote the backbone models as MLM-based methods, Qwen-0.5B, Qwen-1.5B, Qwen-7B throughout the rest of the paper.

3.3 The LODO Results

3.3.1 Results on the Individual Datasets

Tables 1 and 2 show the experimental results on the Amazon Review and MNLI datasets under the LODO setting. From these experimental results, we can observe that: 1) CO4S achieves the best performance on both datasets, obtaining macro-F1 scores of 85.66% on the Amazon dataset and 51.93% on the MNLI dataset, surpassing the second-best model by 1.71% and 2.00%, respectively. 2) When the backbone is frozen, on the Amazon dataset, only ELS and DomCLP outperform the backbone, with imeprovements of just 0.13% and 1.20%, respectively. On the MNLI dataset, only ELS slightly exceeds the backbone by 0.96%. However, these gains come at a substantial cost: the number of parameters in ELS and DomCLP is significantly higher than that of the backbone, increasing by 789.53% and 743.80% on the Amazon dataset, and by 1029.48% and 485.09% on the MNLI dataset, respectively. 3) In addition, CO4S uses only 0.0015M and 0.0023M trainable parameters on the two datasets, respectively, reducing the number of trainable parameters by 99.81% and 99.71% compared to MSCL, the second most parameter-efficient model. Despite this extreme parameter efficiency, CO4S outperforms MSCL by 13.71% and 18.40% in macro-F1 on the respective datasets. From these observations, we can conclude that: 1) PLMs inherently capture rich domain-invariant features, supporting the observation in Fig. 1(a). 2) Complex learning strategies do not necessarily yield better domain-invariant representations; on the contrary, they may disrupt the intrinsic feature structure already present in PLMs, supporting the observation in Figs. 1(b)-1(d). And 3) CO4S effectively models the domain discrepancy between source and target domains (with the highest macro-F1 and the lowest trainable parameters). Remarkably, even without access to target-domain data, our method enhances model generalization, demonstrating its practical efficacy in real-world domain generalization scenarios.

3.3.2 Results on the Cross Datasets

Table 3 shows the experimental results on the cross-dataset setting. From these results, we can observe that: 1) In the cross-dataset setting, where domain discrepancies become more complex, CO4S shows only a marginal improvement over other baselines, outperforming the second-best model DomCLP by just 0.26% on average, yet it achieves this with 99.87% fewer trainable parameters. And 2) compared to the backbone, CO4S demonstrates a stronger ability to capture domain-invariant features, yielding performance gains of 4.86%, 1.51%, 6.72%, and 15.19% on the four datasets, respectively. From these observations, we can conclude that: 1) CO4S can further enhance the generalization capability of PLMs by more effectively capturing domain-invariant features, even when the PLM already exhibits some degree of domain generalization. And 2) although CO4S may underperform compared to certain domain generalization methods in some cross-dataset scenarios, its highly parameter-efficient design makes it significantly more suitable for real-world deployment.

3.4 Ablation Study

Since CO4S is a lightweight, module-free, and hyperparameter-free method, we conduct the following three ablation studies in the ablation section: 1) Unfreezing the backbone parameters during training; 2) Replacing the mean in the computation of $\boldsymbol{c}^{src}$ and $\boldsymbol{c}^{tar}$ with the median (denoted as CO4S (median)); And 3) Using a random offset instead of the domain shift vector $\boldsymbol{d}$ (denoted as CO4S (random)).

Table 4 presents the ablation study results. From these results, we can observe that: 1) Backbones with unfrozen parameters significantly outperform those with frozen parameters in terms of overall performance (Backbone: 75.74% vs. 57.46%; CO4S: 75.59% vs. 62.98%). However, while CO4S slightly decreases performance by 0.14% points in the non-freeze setting, it brings a substantial improvement of 5.51% points when the backbone is frozen. 2) Using a random offset leads to a substantial performance drop (averaging a 34.94% decrease across the 6 datasets); And 3) replacing the mean with the median results in only a minor degradation (0.82%).

These findings suggest that: 1) While fine-tuning pretrained parameters can improve model performance, it also increases the risk of overfitting to the

Table 3: The LODO results on the SST-2, IMDB, SICK, and SNLI datasets with Roberta-base as backbone (%).

Model	SA NTP	NLI NTP	Amazon→SST-2	Amazon→IMDB	MNLI→SICK	MNLI→SNLI	Avg.
PDA	4.1906	6.5560	35.30 ± 11.49	30.80 ± 10.00	EG	EG	-
EAGLE	4.3710	6.7372	35.26 ± 1.62	54.84 ± 13.70	20.64 ± 4.44	6.18 ± 3.13	29.23 ± 18.01
MSCL	0.7890	0.7895	43.66 ± 5.64	73.47 ± 2.53	35.03 ± 1.63	17.24 ± 4.34	42.35 ± 20.34
SwAV	1.2485	1.2493	46.87 ± 10.54	60.14 ± 2.93	18.07 ± 2.29	7.59 ± 0.41	33.17 ± 21.20
ELS	2.3693	4.1449	77.66 ± 1.67	80.13 ± 1.18	46.38 ± 3.68	10.49 ± 0.61	53.67 ± 28.25
DomCLP	1.1172	1.1180	80.34 ± 1.44	**82.56 ± 0.37**	47.64 ± 3.88	**28.67 ± 9.71**	59.80 ± 22.68
Backbone	**0.0015**	**0.0023**	75.89 ± 3.11	80.44 ± 1.58	43.51 ± 6.08	12.16 ± 0.45	53.00 ± 27.54
CO4S	**0.0015**	**0.0023**	**82.53 ± 0.68**	81.95 ± 0.04	**50.22 ± 3.59**	27.35 ± 1.65	**60.51 ± 23.18**
Improve	↓ 99.81%	↓ 99.71%	↑ 2.19	↓ 0.61	↑ 2.58	↓ 1.32	↑ 0.71

Table 4: The ablation study on LODO setting (%). All experiments are implemented using RoBERTa-base. "Improve" refers to the performance gain of CO4S relative to the backbone here.

Model	Freeze	Amazon	MNLI	SST-2	IMDB	SICK	SNLI	Avg.
Backbone	✗	90.35	77.47	90.64	86.69	53.95	55.33	75.74
CO4S (mean)	✗	90.65	77.54	90.84	87.57	59.75	47.21	75.59
Improve		↑ 0.30	↑ 0.07	↑ 0.20	↑ 0.88	↑ 5.80	↓ 8.12	↓ 0.14
Backbone	✓	83.82	48.97	75.89	80.44	43.51	12.16	57.46
CO4S (median)	✓	85.55	51.41	82.16	81.56	49.27	24.76	62.45
CO4S (random)	✓	41.98	20.90	49.71	35.30	14.83	7.25	28.33
CO4S (mean)	✓	85.66	51.93	82.53	81.95	50.22	27.35	63.27
Improve		↑ 1.84	↑ 2.96	↑ 6.65	↑ 1.51	↑ 6.72	↑ 15.19	↑ 5.81

source domain. In contrast, our method enhances generalization performance without any fine-tuning of the pretrained backbone, especially in practical scenarios where fine-tuning large-scale models is often infeasible. 2) The direction of the shift is crucial to the CO4S's performance; And 3) The feature distributions extracted by the PLM are approximately symmetric to some extent, and the difference between the mean and median vectors is approximately orthogonal to the classifier weights, indicating that this discrepancy resides primarily in non-discriminative features (see Appendix H for detailed analysis).

3.5 Model Scale Analysis

In this section, we evaluate the models' scale from 3 perspectives: 1) the number of trainable parameters; 2) the storage size of trainable parameters; and 3) peak GPU memory usage during training.

Table 5 shows the experimental results of models' scale with "Kitchen" as the target domain. From these results, we can observe that ***our method introduces only 7.83KB of trainable parameters, yet achieves an average improvement of 5.51% in macro-F1 across 6 datasets***. Moreover, even **for a 7B LLM, only 29.68KB of additional parameters are required to enhance the model's generalization ability.** In addition, the peak GPU memory usage experiments demonstrate that our method does not require expensive hardware, only **0.52 GB of GPU memory** is needed to train our model.

3.6 Domain Discrepancy Analysis

We conduct domain discrepancy analysis by measuring the Maximum Mean Discrepancy (MMD) (Gretton et al., 2012) between source and target domains. This allows us to evaluate whether our method effectively mitigates domain shifts. We conduct experiments on the SA task under both in-dataset and cross-dataset settings. For the in-dataset setup, we use the Amazon dataset with book, dvd, and electronics as source domains and kitchen as the target domain. For the cross-dataset setup, we use the entire Amazon dataset as the source and SST-2 as the target. We compute the MMD between each source-target pair and report the average MMD as the final domain discrepancy, i.e.

$$MS = \frac{1}{N^{src}} \sum_{i=1}^{N^{src}} \text{MMD}(\boldsymbol{h}_i^{src}, \boldsymbol{h}^{tar}),$$

where MS is the overall MMD score; $\text{MMD}(a, b)$ computes the MMD score between a and b; N^{src} is the number of source domains.

All experiments are conducted using the same random seed for consistency. Table 6 shows the experimental results. From these results, we can observe that CO4S achieves the lowest average

Table 5: The models' scale analysis. We train each model using the kitchen in the Amazon dataset as the target domain. Bold values indicate the best scores.

Model	Number of trainable parameters (M)	File sizes (KB)	Peak GPU memory usage (GB)
PDA	2.4165	9,448.80	0.9183
EAGLE	2.5969	10,155.16	0.6482
SwAV	1.2485	4,882.49	0.5394
RoBERTa-base	**0.0015**	**7.83**	**0.5207**
Qwen-0.5B	**0.0018**	**8.69**	1.9601
Qwen-1.5B	**0.0031**	**13.69**	5.9634
Qwen-7B	**0.0082**	**29.68**	26.95

Table 6: The domain discrepancy analysis results. Bold values indicate the best scores.

Domain	RoBERTa-base	RoBERTa-base (CO4S)
Kitchen	0.0835	**0.0301**
SST-2	0.2888	**0.0396**

MMD scores in both settings, with 0.0301 in the individual-dataset setting and 0.0396 in the cross-dataset setting. Compared to the backbone, CO4S decreases by 0.0534 and 0.2492 respectively in the two settings. From these observations, we can conclude that although pretrained language models have already captured rich domain-invariant features, our method further reduces the domain discrepancy beyond what the pretrained models achieve.

4 Related Work

Domain generalization aims to train the model on multiple source domains, learn domain-invariant features, and then transfer knowledge to unseen target domains. Existing text domain generalization methods have explored various complex techniques to enhance model generalization. PDA (Jia and Zhang, 2022), MIL (Jia and Zhang, 2023), EAGLE (Bhattacharjee et al., 2024), ELS (Zhang et al., 2023), and DAAL (Chen et al., 2025) adopt an adversarial training strategy to learn domain-invariant features. MSCL (Tan et al., 2022), EAGLE (Bhattacharjee et al., 2024), and DomCLP (Lee et al., 2025) attempt to leverage contrastive learning to capture the similarity across different domains, thereby improving the model's ability to discriminate between classes from various domains. Furthermore, Jin et al. (Jin et al., 2020), Nguyen et al. (Nguyen et al., 2022), and Wei et al. (Wei et al., 2025) employed the method of feature alignment to align the features from different domains as closely as possible during the training phase. Moreover, TACIT (Song et al., 2024) uses a teacher-student framework to filter samples based on difficulty levels. DFGN (Ren et al., 2023) uses Shannon entropy to measure the domain uncertainty of model outputs. PADA (Ben-David et al., 2022) requires generating an additional task- and domain-specific prompt during testing before performing downstream inference. All these methods explore different strategies to learn domain-invariant features. However, they overlook the fact that PLMs have already learned domain-invariant features from large-scale datasets during pre-training.

5 Conclusion

In this paper, we introduce CO4S, a simple yet effective method for domain generalization that leverages the intrinsic domain-invariant properties of pretrained language models. Unlike existing approaches that require complex training strategies and risk overfitting the source domain, CO4S operates solely by training a lightweight classifier on the source domain and applying a test-time domain shift correction. Our method requires no updates to the backbone and introduces negligible computational overhead. Extensive experiments across 6 benchmark datasets show that CO4S consistently improves generalization performance, achieving a 5.51% average macro-F1 gain with only 7.83KB of trainable parameters. Additionally, CO4S demonstrates superior training efficiency, especially with large language models. These results highlight the promise of rethinking domain generalization from a lightweight, inference-time perspective.

6 Limitations

While CO4S is simple and effective, it comes with several limitations. 1) CO4S is built upon the assumption that PLMs can learn rich textual semantic representations and thus are capable of capturing domain-invariant features from text. However,

in domains such as CV, where inter-domain discrepancies are often much more pronounced and structural (e.g., in pixel-level distributions or visual styles), CO4S, which is a linear adaptation method, may face significant limitations. 2) Our method assumes access to sufficient unlabeled target domain samples at test time. In scenarios with scarce or streaming target data, the effectiveness of CO4S may degrade. In addition, CO4S imposes certain requirements on the pre-trained language model. If the sentence embeddings produced by the language model are unevenly distributed, the performance of our method may degrade. For more details, please refer to Appendix J. 3) CO4S assumes that the source and target domains are moderately aligned in feature space. When the source and target data are small or highly imbalanced, CO4S may underperform, as observed in the extended experiments reported in Appendix K. And 4) CO4S can be used as a plug-in module and readily applied to other models. However, it relies on the assumption that pretrained language models already learn rich domain-invariant features. When alternative learning strategies (such as adversarial approaches like EAGLE) are applied to optimize the representations of pretrained models further, they may inadvertently disrupt these intrinsic features, thereby degrading the effectiveness of CO4S. We provide empirical evidence for this behavior in Appendix G.

7 Acknowledgements

We acknowledge that in our writing, Large Language Model is used only for polishment.

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR.

Eyal Ben-David, Nadav Oved, and Roi Reichart. 2022. PADA: example-based prompt learning for on-the-fly adaptation to unseen domains. *Transactions of the Association for Computational Linguistics*, 10:414–433.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine Learning*, 79(1-2):151–175.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. 2006. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems*, pages 137–144. MIT Press.

Amrita Bhattacharjee, Raha Moraffah, Joshua Garland, and Huan Liu. 2024. EAGLE: A domain generalization framework for ai-generated text detection. *CoRR*, abs/2403.15690.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*. The Association for Computational Linguistics.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. 2020. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*.

Jianting Chen, Ling Ding, Yunxiao Yang, Zaiyuan Di, and Yang Xiang. 2025. Domain adversarial active learning for domain generalization classification. *IEEE Transactions on Knowledge and Data Engineering*, 37(1):226–238.

Saurab Chhachhi and Fei Teng. 2023. On the 1-wasserstein distance between location-scale distributions and the effect of differential privacy. *Preprint*, arXiv:2304.14869.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 81 others. 2024. Deepseek-v3 technical report. *CoRR*, abs/2412.19437.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and GPT-2 embeddings. 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 2019*, pages 55–65. Association for Computational Linguistics.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019. Towards understanding linear word analogies.

In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Volume 1: Long Papers*, pages 3253–3262. Association for Computational Linguistics.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. 2012. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773.

Xu Guo and Han Yu. 2022. On the domain adaptation and generalization of pretrained language models: A survey. *CoRR*, abs/2211.03154.

Chen Jia and Yue Zhang. 2022. Prompt-based distribution alignment for domain generalization in text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022*, pages 10147–10157. Association for Computational Linguistics.

Chen Jia and Yue Zhang. 2023. Memory-based invariance learning for out-of-domain text classification. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023*, pages 1635–1647. Association for Computational Linguistics.

Xin Jin, Cuiling Lan, Wenjun Zeng, and Zhibo Chen. 2020. Feature alignment and restoration for domain generalization and adaptation. *CoRR*, abs/2006.12009.

Elena Kochkina, Maria Liakata, and Arkaitz Zubiaga. 2018a. All-in-one: Multi-task learning for rumour verification. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018*, pages 3402–3413. Association for Computational Linguistics.

Elena Kochkina, Maria Liakata, and Arkaitz Zubiaga. 2018b. All-in-one: Multi-task learning for rumour verification. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018*, pages 3402–3413. Association for Computational Linguistics.

Jin-Seop Lee, Noo-Ri Kim, and Jee-Hyong Lee. 2025. Domclp: Domain-wise contrastive learning with prototype mixup for unsupervised domain generalization. In *AAAI 2025*, pages 18119–18127. AAAI Press.

Chen Ling, Xujiang Zhao, Jiaying Lu, Chengyuan Deng, Can Zheng, Junxiang Wang, Tanmoy Chowdhury, Yun Li, Hejie Cui, Xuchao Zhang, Tianjiao Zhao, Amit Panalkar, Dhagash Mehta, Stefano Pasquali, Wei Cheng, Haoyu Wang, Yanchi Liu, Zhengzhang Chen, Haifeng Chen, and 5 others. 2024. Domain specialization as the key to make large language models disruptive: A comprehensive survey. *Preprint*, arXiv:2305.18703.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*. OpenReview.net.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014*, pages 216–223. European Language Resources Association (ELRA).

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Workshop Track Proceedings*.

Thuan Nguyen, Boyang Lyu, Prakash Ishwar, Matthias Scheutz, and Shuchin Aeron. 2022. Trade-off between reconstruction loss and feature alignment for domain generalization. In *21st IEEE International Conference on Machine Learning and Applications, ICMLA 2022*, pages 794–801. IEEE.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Gabriel Peyré and Marco Cuturi. 2019. Computational optimal transport. *Foundations and Trends in Machine Learning*, 11(5-6):355–607.

Emily Reif, Ann Yuan, Martin Wattenberg, Fernanda B. Viégas, Andy Coenen, Adam Pearce, and Been Kim. 2019. Visualizing and measuring the geometry of BERT. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 8592–8600.

He Ren, Jun Wang, Zhongkui Zhu, Juanjuan Shi, and Weiguo Huang. 2023. Domain fuzzy generalization networks for semi-supervised intelligent fault diagnosis under unseen working conditions. *Mechanical Systems and Signal Processing*, 200:110579.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013*, pages 1631–1642. ACL.

Rui Song, Fausto Giunchiglia, Yingji Li, Mingjie Tian, and Hao Xu. 2024. TACIT: A target-agnostic feature disentanglement framework for cross-domain text classification. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024*, pages 18999–19007. AAAI Press.

Qingyu Tan, Ruidan He, Lidong Bing, and Hwee Tou Ng. 2022. Domain generalization for text classification with memory-based supervised contrastive learning. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022*, pages 6916–6926. International Committee on Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019*. OpenReview.net.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip S. Yu. 2023a. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):8052–8072.

Xu Wang, Dezhong Peng, Peng Hu, Yunhong Gong, and Yong Chen. 2023b. Cross-domain alignment for zero-shot sketch-based image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(11):7024–7035.

Wei Wei, Zixiong Li, Jing Yan, Mingwen Shao, and Lin Li. 2025. Indirect alignment and relationship preservation for domain generalization. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2025*, pages 2054–2062. ijcai.org.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 40 others. 2024. Qwen2 technical report. *CoRR*, abs/2407.10671.

Yifan Zhang, Xue Wang, Jian Liang, Zhang Zhang, Liang Wang, Rong Jin, and Tieniu Tan. 2023. Free lunch for domain adversarial training: Environment label smoothing. In *The Eleventh International Conference on Learning Representations, ICLR 2023*. OpenReview.net.

A Theoretical Analysis

In this section, we provide a theoretical analysis of CO4S by building upon the classical risk upper-bound theory (Ben-David et al., 2006; Wang et al., 2023a) from Domain Generalization (DG) and Domain Adaptation (DA).

A.1 Preliminaries and Definitions

Let the input space be $\mathcal{X}$, the representation space be $\mathcal{Z}$, and the label space be $\mathcal{Y}$. Given a fixed PLM f, there has

$$f : \mathcal{X} \to \mathcal{Z}.$$

In addition, given a trainable classifier f_c, there has

$$f_c : \mathcal{Z} \to \mathcal{Y}.$$

Let the source distribution be $\mathcal{P}_X^{src}$ and the target distribution be $\mathcal{P}_X^{tar}$, and their push-forward distribution in the representation space be

$$\mathcal{P}_Z^{src} \coloneqq f_\sharp \mathcal{P}_X^{src}, \mathcal{P}_Z^{tar} \coloneqq f_\sharp \mathcal{P}_X^{tar}.$$

Furthermore, CO4S introduces a domain shift vector in the test phase, which is defined as

$$\boldsymbol{d} \coloneqq \boldsymbol{c}^{src} - \boldsymbol{c}^{tar}.$$

Definition 1 (Average Risk Estimation Error Bound)**.** *According to (Wang et al., 2023a), the average risk over all possible target domains of classifier h is defined as*

$$\epsilon(h) \coloneqq \mathbb{E}_{P_{XY} \sim \mathcal{P}} \mathbb{E}_{(x,y) \sim P_{XY}} [l(h(P_X, \boldsymbol{x}), y)], \tag{4}$$

where l is a loss funcion on $\mathcal{Y}$; $\mathcal{P}$ is a hyper-distribution, that $\mathcal{P}_{XY}^{tar} \sim \mathcal{P}$ and $\mathcal{P}_{XY}^{src} \sim \mathcal{P}$; (x, y) is the finite sample set samples from each distribution.

Definition 2 (Domain Adaptation Error Bound)**.** *According to (Ben-David et al., 2006, 2010; Wang et al., 2023a), the domain adaptation error bound is defined as: for any $h \in \mathcal{H}$ and $\delta \in (0, 1)$, there is at least $1 - \delta$ probability*

$$\begin{aligned} \epsilon^{tar}(h) \leq & \epsilon^{src}(h) + \hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}^{src}, \mathcal{U}^{tar}) + \lambda_{\mathcal{H}} \\ & + 4\sqrt{\frac{2d\log(2n) + \log(2/\delta)}{n}}, \end{aligned} \tag{5}$$

where d is the Vapnik-Chervonenkis dimension of the hypothesis space $\mathcal{H}$; $\mathcal{U}^{src}$ and $\mathcal{U}^{tar}$ are unlabeled samples of size n from source and target domains; $\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}^{src},\mathcal{U}^{tar})$ is the estimate of $d_{\mathcal{H}\Delta\mathcal{H}}(P_X^{src}, P_X^{tar})$ on the two sets of finite data samples, and $d_{\mathcal{H}\Delta\mathcal{H}}(P_X^{src}, P_X^{tar}) := \sup_{h,h'\in\mathcal{H}} |\epsilon^{src}(h,h') - \epsilon^{tar}(h,h')|$ is the $\mathcal{H}\Delta\mathcal{H}$-divergence; $\lambda_{\mathcal{H}}$ is an ideal joint error related to the hypothesis space $\mathcal{H}$, which characterizes the minimum possible error of a hypothesis that performs well simultaneously in both the source domain and the target domain.

Definition 3 (Push-forward Measure)**.** *According to (Peyré and Cuturi, 2019), the push-forward measure is defined as follows. Let $(\mathcal{X}, \mathcal{A})$ and $(\mathcal{Z}, \mathcal{B})$ be two measurable spaces; $f : \mathcal{X} \to \mathcal{Z}$ be a measurable mapping; and P_X be a probability measure defined on $(\mathcal{X}, \mathcal{A})$. Then the push-forward measure of P_X induced by f on $\mathcal{Z}$, denoted by $P_Z = f_\sharp P_X$, is defined as follows: for any measurable set $A \in \mathcal{B}$,*

$$P_Z(A) \coloneqq (f_\sharp P_X)(A) = P_X(f^{-1}(A)), \quad (6)$$

where $f^{-1}(A) \coloneqq \{x \in \mathcal{X} : f(x) \in A\}$ is the inverse image of a set.

A.2 Analysis of the risk upper bound of CO4S

Lemma 1 (Source Domain Risk Stability Under Fixed Representation)**.** *Fixing the backbone maintains a comparably low source domain risk due to the linear separability of PLM representations, while reducing the hypothesis space complexity.*

Proof. According to Definition 1, we can define the source domain risk be

$$\epsilon_{src}(h) = \mathbb{E}_{(x,y)\sim\mathcal{P}^{src}}\left[l(f_c(f(x)), y)\right],$$

where $h \coloneqq f_c \circ f$ is the hypothesis function.

We analyze the risk from the perspective of the generalization bound. The expected risk on the source domain $\epsilon_{src}(h)$ is bounded by the empirical risk $\hat{\epsilon}_{src}(h)$ and a complexity term $\mathcal{C}(\mathcal{H})$:

$$\epsilon_{src}(h) \le \hat{\epsilon}_{src}(h) + \mathcal{O}\left(\sqrt{\frac{d_{VC}}{n}}\right)$$

where d_{VC} is the VC-dimension of the hypothesis space.

- **Empirical Risk**: Based on the proven linear separability of PLM representations (Ethayarajh, 2019; Reif et al., 2019), the pre-trained features $f(\cdot)$ are sufficiently discriminative. Thus, a linear classifier f_c can achieve near-zero empirical risk, i.e., $\hat{\epsilon}_{src}(f_c \circ f_{fixed}) \approx \hat{\epsilon}_{src}(f_c \circ f_{fine-tuned}) \approx 0$.
- **Complexity**: Full fine-tuning updates all parameters $\theta_f + \theta_c$, resulting in a hypothesis space with extremely high dimension d_{VC}^{full}. In contrast, CO4S fixes θ_f and only updates θ_c, yielding a significantly smaller VC-dimension $d_{VC}^{CO4S} \ll d_{VC}^{full}$.

While full fine-tuning might marginally reduce training loss, it risks increasing the bound on $\epsilon_{src}(h)$ due to the complexity penalty (overfitting). By fixing the backbone, CO4S maintains a comparable empirical risk while minimizing the complexity term, thereby ensuring the source domain expected risk $\epsilon_{src}(h)$ does not degrade (and is likely more stable).

□

Lemma 2 (The Representation Translation of CO4S is Equivalent to the Distribution Push-forward)**.** *The operations performed by CO4S on the target domain representation*

$$\boldsymbol{h}^{tar} \mapsto \boldsymbol{h}^{tar} + \boldsymbol{d}$$

are equivalent to constructing a new target domain distribution in the representation space

$$\tilde{\mathcal{P}}_Z^{tar} \coloneqq T_d \sharp \mathcal{P}_Z^{tar},$$

where $T_d = \boldsymbol{h}^{tar} + \boldsymbol{d}$; and $\sharp$ represents the push-forward measurement.

Proof. According to Definition 3, for any measurable set $A \subset \mathcal{Z}$, there has

$$\begin{aligned}\tilde{\mathcal{P}}_Z^{tar}(A) &= \mathcal{P}_Z^{tar}(T_d^{-1}(A)) \\ &= \mathcal{P}_Z^{tar}(\{\boldsymbol{h}^{tar} : \boldsymbol{h}^{tar} + \boldsymbol{d} \in A\}).\end{aligned}$$

This precisely corresponds to applying a uniform shift $\boldsymbol{d}$ to the representations of all target domains. Consequently, CO4S does not modify the classifier or the training procedure; instead, it implicitly constructs a new target-domain representation distribution at test time. □

Assumption 1 (Additive Domain Shift)**.** *Following the linearity of semantic shifts in PLMs (Ethayarajh, 2019), we assume the domain discrepancy in the deep representation space can be primarily*

modeled as an additive transformation. Specifically, the target distribution $\mathcal{P}_Z^{tar}$ is an affine translation of the source distribution $\mathcal{P}_Z^{src}$ subject to a noise term ξ: $Z^{tar} = Z^{src} + \boldsymbol{d} + \xi$, where ξ represents higher-order discrepancies (e.g., covariance shift) which are assumed to be sub-dominant compared to $\boldsymbol{d}$.

Proposition 3 (CO4S Minimizes the First-Order Component of Distribution Discrepancy)**.** *Under Assumption 1, CO4S reduces the upper bound of the Wasserstein-1 distance composed of mean shift and shape discrepancy between the source and target domains by eliminating the discrepancy in the first moment (mean), thereby tightening the $\mathcal{H}\Delta\mathcal{H}$-divergence bound.*

Proof. According to the property of the Wasserstein-1 distance (Arjovsky et al., 2017; Chhachhi and Teng, 2023), for any two distributions P and Q, the distance is lower-bounded by the distance between their means:

$$W_1(P, Q) \geq ||\mathbb{E}_P[z] - \mathbb{E}_Q[z]||.$$

The original discrepancy between source and target is $W_1(\mathcal{P}_Z^{src}, \mathcal{P}_Z^{tar}) \geq ||\boldsymbol{c}^{src} - \boldsymbol{c}^{tar}||$.

In CO4S, we apply the shift vector $\boldsymbol{d} = \boldsymbol{c}^{src} - \boldsymbol{c}^{tar}$ to the target representations, constructing a new distribution $\tilde{\mathcal{P}}_Z^{tar}$. The mean of this new distribution is:

$$\begin{aligned}\mathbb{E}_{z\sim\tilde{\mathcal{P}}_Z^{tar}}[z] &= \mathbb{E}_{z\sim\mathcal{P}_Z^{tar}}[z + d] \\ &= \boldsymbol{c}^{tar} + (\boldsymbol{c}^{src} - \boldsymbol{c}^{tar}) \\ &= \boldsymbol{c}^{src}\end{aligned}$$

Consequently, the distance between the means becomes:

$$||\mathbb{E}_{\mathcal{P}_Z^{src}}[z] - \mathbb{E}_{\tilde{\mathcal{P}}_Z^{tar}}[z]|| = ||\boldsymbol{c}^{src} - \boldsymbol{c}^{src}|| = 0.$$

By eliminating the distance between centroids, CO4S minimizes the translation component of the Wasserstein distance.

Let Z^{src} and Z^{tar} be random variables following distributions $\mathcal{P}_Z^{src}$ and $\mathcal{P}_Z^{tar}$, respectively. If we decompose the domain shift into a mean shift and a shape shift (centered distributions), for the original distribution, we have

$$Z^{src} = \hat{Z}_Z^{src} + \boldsymbol{c}^{src}, Z^{tar} = \hat{Z}_Z^{tar} + \boldsymbol{c}^{tar}, \quad (7)$$

where $\mathbb{E}[\hat{Z}_Z^{src}] = \mathbb{E}[Z^{tar}] = 0$.

By applying the triangle inequality of norms $\|a+b\| \leq \|a\| + \|b\|$, we can derive

$$\begin{aligned}&\|Z^{src} - Z^{tar}\| \\ =&\|(\hat{Z}^{src} + \boldsymbol{c}^{src}) - (\hat{Z}^{tar} + \boldsymbol{c}^{tar})\| \\ =&\|(\boldsymbol{c}^{src} - \boldsymbol{c}^{tar}) + (\hat{Z}^{src} - \hat{Z}^{tar})\| \\ \leq&\|\boldsymbol{c}^{src} - \boldsymbol{c}^{tar}\| + \|\hat{Z}^{src} - \hat{Z}^{tar}\|.\end{aligned} \quad (8)$$

According to the definition of Wasserstein-1 distance $W_1(P, Q) = \inf_{\gamma\in\Pi(Q,P)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|]$, taking the expectation on both sides of Eq. 8 yields the Wasserstein-1 distance:

$$\begin{aligned}W_1(\mathcal{P}_Z^{src}, \mathcal{P}_Z^{tar}) \leq&\|c^{src} - c^{tar}\| \\ &+ W_1(\hat{\mathcal{P}}_Z^{src}, \hat{\mathcal{P}}_Z^{tar}).\end{aligned} \quad (9)$$

Next, for CO4S, based on the push-forward distribution, we can express the target domain centered distribution as

$$\begin{aligned}\tilde{Z}^{tar} &= Z^{tar} + \boldsymbol{d} \\ &= Z^{tar} + \boldsymbol{c}^{src} - \boldsymbol{c}^{tar} \\ &= \hat{Z}^{tar} + \boldsymbol{c}^{tar} + \boldsymbol{c}^{src} - \boldsymbol{c}^{tar} \\ &= \hat{Z}^{tar} + \boldsymbol{c}^{src}.\end{aligned} \quad (10)$$

By substituting Equation 10 into Equations 8 and 9, we obtain

$$\begin{aligned}W_1(\mathcal{P}_Z^{src}, \tilde{\mathcal{P}}_Z^{tar}) \leq&\|\boldsymbol{c}^{src} - \boldsymbol{c}^{src}\| \\ &+ W_1(\hat{\mathcal{P}}_Z^{src}, \hat{\mathcal{P}}_Z^{tar}) \\ =&W_1(\hat{\mathcal{P}}_Z^{src}, \hat{\mathcal{P}}_Z^{tar}).\end{aligned} \quad (11)$$

Based on the above results, we observe that CO4S effectively eliminates the first-order discrepancy term.

Moreover, assuming the classifier f_c is Lipschitz continuous with constant L, the domain divergence bounds become

$$\begin{aligned}d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{P}_Z^{src}, \mathcal{P}_Z^{tar}) &\leq 2L \cdot W_1(\mathcal{P}_Z^{src}, \mathcal{P}_Z^{tar}). \\ d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{P}_Z^{src}, \tilde{\mathcal{P}}_Z^{tar}) &\leq 2L \cdot W_1(\mathcal{P}_Z^{src}, \tilde{\mathcal{P}}_Z^{tar}).\end{aligned}$$

Since $W_1(\mathcal{P}_Z^{src}, \tilde{\mathcal{P}}_Z^{tar})$ under CO4S is significantly smaller than that of the original feature distributions $W_1(\mathcal{P}_Z^{src}, \mathcal{P}_Z^{tar})$, the $d_{\mathcal{H}\Delta\mathcal{H}}$ is reduced accordingly. Consequently, the upper bound on the target risk is strictly tightened.

□

Proposition 4 (CO4S tightens the risk upper bound for the target domain)**.** *Under the covariate shift assumption, CO4S tightens the theoretical upper*

bound on the target risk for domain generalization by reducing the difference term representing the distribution between the source domain and the target domain.

Proof. According to Definition 2, the target risk is upper-bounded by the sum of the source risk $\epsilon_{src}(h)$, a distributional divergence term $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{P}_Z^{src}, \tilde{\mathcal{P}}_Z^{tar})$, the ideal joint risk $\lambda_{\mathcal{H}}$, and a consistent convergent generalization error term based on VC dimension $4\sqrt{\frac{2d\log(2n)+\log(2/\delta)}{n}}$.

We analyze how CO4S tightens this bound term by term:

- **Complexity and Source Risk**: As established in Lemma 1, CO4S significantly reduces the hypothesis space complexity (VC-dimension d) by freezing the backbone parameters compared to full fine-tuning ($d_{CO4S} \ll d_{FT}$). This directly minimizes the complexity term ($4\sqrt{\frac{2d\log(2n)+\log(2/\delta)}{n}}$) in Eq. 5. Consequently, the upper bound on the expected source risk $\epsilon_{src}(h)$ is tightened, preventing the overfitting often seen in over-parameterized fine-tuning.

- **Domain Divergence**: Based on Proposition 3, CO4S explicitly eliminates the first-order moment discrepancy between domains. This reduction in the Wasserstein-1 distance translates to a lower $\mathcal{H}\Delta\mathcal{H}$-divergence $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{P}_Z^{src}, \tilde{\mathcal{P}}_Z^{tar})$, further tightening the bound.

- **Ideal Joint Error**: Since the pre-trained backbone f already encodes rich semantic features, a lightweight classifier f_c is sufficient to achieve low error, ensuring that the ideal joint error $\lambda_{\mathcal{H}}$ remains low.

Therefore, by simultaneously minimizing the distribution divergence and the model complexity term without degrading the ideal error, CO4S tightens the theoretical upper bound on the target domain risk. □

B Evaluation Settings

B.1 Datasets

The detailed statistical analysis of the dataset is shown in Table 7.

Table 7: The details of the experimental datasets.

Task	Dataset	Domains	Train	Test
SA	Amazon	book	2,000	2,000
		dvd	2,000	2,000
		electronics	2,000	2,000
		kitchen	2,000	2,000
	IMDB	movie	-	25,000
	SST-2	movie	-	1,821
NLI	MNLI	fiction	2,547	2,547
		government	2,541	2,541
		slate	2,605	2,605
		telephone	2,754	2,754
		travel	2,541	2,541
	SNLI	general	-	4,097
	SICK	image&video	-	4,906

B.2 Generalization Settings

In the appendix, in addition to the LODO setting, to further demonstrate the generality of our method, we consider two additional experimental setups: single-domain generalization (SDG) and partial-domain generalization (PDG). The precise definitions of these two settings are as follows:

Single-Domain Generalization (SDG). In this setting, we only adopt 1 domain as the source domain ($|\mathcal{D}^{src}| = 1$) and test the model on 1 target domain ($|\mathcal{D}^{tar}| = 1$), $\mathcal{D}^{src} \neq \mathcal{D}^{tar}$.

Partial-Domain Generalization (PDG). In this setting, we adopt P ($(1 < P < N - 1)$) domains as the source domains and test the model on the rest target domains ($|\mathcal{D}^{tar}| = N - P$), $\mathcal{D}^{src} \neq \mathcal{D}^{tar}$.

C Comparing with Other Backbones

To verify that the performance gains of CO4S are independent of the choice of backbone, we further compare the performance of these methods across different backbone architectures. Tables 8, 9 and 10 show the experimental results on the Amazon, MNLI, and cross datasets with BERT-base as backbone, while Tables 11, 12 and 13 present the results with DistilRoBERTa-base as backbone. From these experimental results, we can conclude that CO4S consistently outperforms the second-best model by 1.50%, 0.99%, 6.54%, 1.08%, 1.82%, and 5.14% in macro-F1 across 6 experimental settings, respectively. These results demonstrate that CO4S is not limited by the choice of PLM, even when using a distilled and more compact model like DistilRoBERTa-base, CO4S still achieves the best performance.

Table 8: The LODO results on the Amazon dataset with BERT-base as backbone (%).

Model	NTP	BDE→K	BDK→E	BEK→D	DEK→B	Avg.
PDA	2.3967	76.63 ± 2.16	77.85 ± 1.97	75.26 ± 0.91	73.49 ± 1.22	75.80 ± 1.62
EAGLE	2.5969	79.34 ± 0.14	75.68 ± 0.45	76.42 ± 0.26	70.65 ± 3.89	75.52 ± 3.13
MSCL	0.7887	73.35 ± 2.04	73.13 ± 1.57	70.75 ± 1.62	72.75 ± 0.37	72.49 ± 1.03
SwAV	1.2485	74.24 ± 2.22	74.60 ± 1.15	71.51 ± 1.14	68.79 ± 2.89	72.29 ± 2.34
ELS	1.1858	81.29 ± 1.75	80.38 ± 0.76	77.49 ± 0.74	75.61 ± 3.44	78.69 ± 2.27
DomCLP	1.1172	73.33 ± 8.62	74.00 ± 3.70	71.29 ± 4.45	76.24 ± 0.33	73.72 ± 1.76
Backbone	**0.0015**	81.23 ± 1.12	79.62 ± 1.24	77.94 ± 0.66	74.65 ± 1.39	78.36 ± 2.44
CO4S	**0.0015**	**82.23 ± 0.45**	**81.81 ± 0.70**	**78.02 ± 0.38**	**78.88 ± 0.85**	**80.23 ± 1.82**
Improve	↓ 99.81%	↑ 0.94	↑ 1.43	↑ 0.07	↑ 2.65	↑ 1.54

Table 9: The LODO results on the MNLI dataset with BERT-base as backbone (%).

Model	NTP	FGST→T*	FGST*→T	FGTT*→G	FSTT*→G	GSTT*→F	Avg.
PDA	4.1708	EG	EG	EG	EG	EG	-
EAGLE	3.7812	36.86 ± 4.29	38.09 ± 3.56	36.21 ± 3.86	36.47 ± 2.78	32.19 ± 4.83	35.96 ± 1.99
MSCL	0.7893	30.70 ± 0.75	31.42 ± 1.23	34.72 ± 0.29	31.96 ± 1.80	33.70 ± 1.54	32.50 ± 1.49
SwAV	1.2493	32.15 ± 1.85	34.41 ± 2.59	36.19 ± 1.95	37.13 ± 1.07	33.98 ± 2.10	34.77 ± 1.74
ELS	2.3701	45.42 ± 0.57	41.42 ± 2.33	44.06 ± 0.57	46.22 ± 1.49	42.86 ± 0.63	44.00 ± 1.73
DomCLP	1.1180	18.71 ± 1.81	18.07 ± 1.37	23.03 ± 4.12	17.27 ± 0.62	16.89 ± 0.42	18.80 ± 2.21
Backbone	**0.0023**	43.54 ± 4.09	41.38 ± 3.06	44.35 ± 0.73	45.94 ± 1.21	41.27 ± 1.71	43.30 ± 1.78
CO4S	**0.0023**	**47.51 ± 0.77**	**42.74 ± 0.93**	**45.33 ± 0.67**	**47.08 ± 1.29**	**43.42 ± 0.64**	**45.22 ± 1.22**
Improve	↓ 99.71%	↑ 2.09	↑ 1.32	↑ 0.98	↑ 0.86	↑ 0.56	↑ 1.22

D Integrating CO4S into Large Language Models

Furthermore, in the experiments reported in Tables 14 and 15, we also employ large language models as backbone networks to evaluate whether CO4S remains effective when applied to larger generative models. We choose Qwen2.5-0.5B-Instruct, Qwen2.5-1.5B-Instruct, and Qwen2.5-7B-Instruct as the backbone LLM.

The results show that, on the Amazon dataset, integrating CO4S improves the performance of the 0.5B, 1.5B, and 7B models by 0.82%, 0.93%, and 0.13% in macro-F1, respectively. On the MNLI dataset, the same integration yields gains of 0.34%, 0.99%, and 0.51% for the three model sizes, respectively. These findings indicate that CO4S is not only effective for models pre-trained with MLM but also enhances the generalization capability of generative large language models on classification tasks.

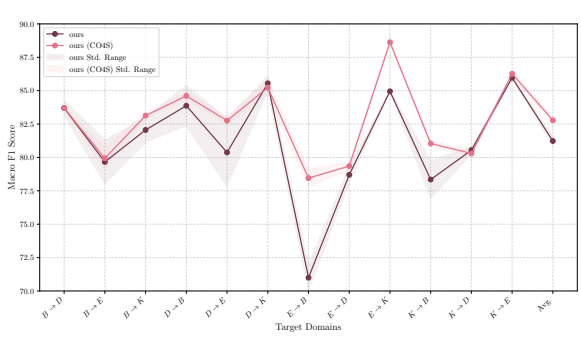
E SDG Results

In this section, we report the SDG results on the six datasets. SDG is a particularly challenging setting in domain generalization, as the model has access to data from only a single source domain during training and is expected to generalize to an unseen target domain[9]. Figs. 3, 4, 5, and 6 show the SDG results for RoBERTa-base, Qwen-0.5B, Qwen-1.5B, and Qwen-7B on six datasets, respectively. From these experimental results, we can observe that CO4S consistently improves model performance on average. Moreover, compared to the backbone, CO4S exhibits lower standard deviation, indicating that it achieves better robustness under single-domain generalization.

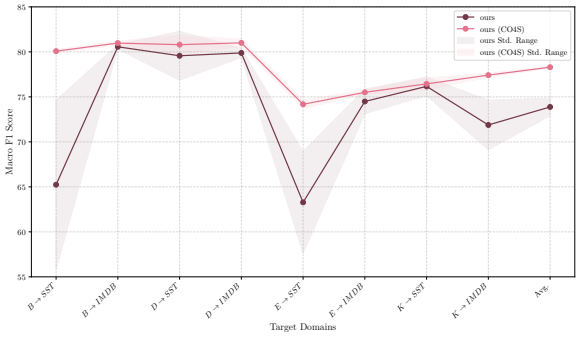
F PDG Results

In this section, we report the PDG results on the six datasets. We evaluate CO4S on 2 source domains for the Amazon dataset; 2 and 3 source domains for the MNLI, SST-2 and IMDB datasets; 2, 3, and 4 source domains for the SICK and SNLI datasets. Figs. 7-10 show the PDG results for RoBERTa-base, Qwen-0.5B, Qwen-1.5B, and Qwen-7B on six datasets, respectively. Experimental results show that CO4S achieves superior average performance and lower standard deviation. Furthermore, based on the SDG (Appendix E) and distribution visualization (Section J) experiments, we observe that CO4S tends to yield better performance when the data within domains is relatively evenly distributed. For example, in the Amazon dataset, when

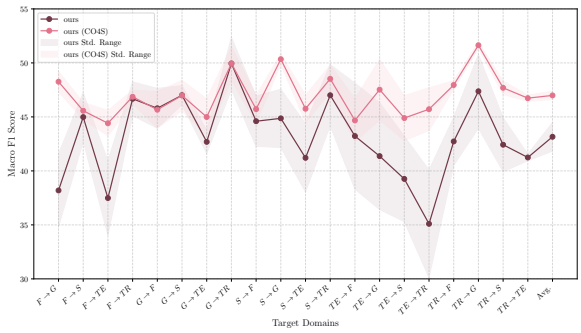
[9] In all experiments presented in the appendix, we use "TE" as the abbreviation for telephone and "TR" for travel in the MNLI dataset.



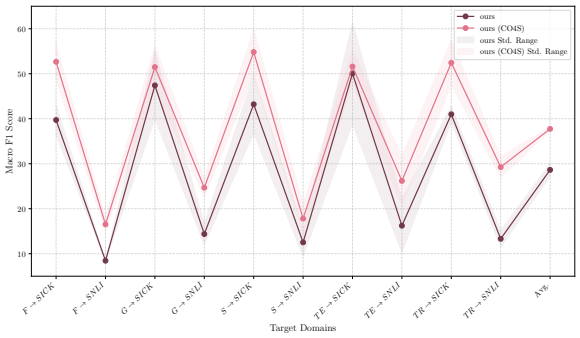
(a) Source and target domains: amazon.



(b) Source domain: amazon; target domains: SST-2 and IMDB.

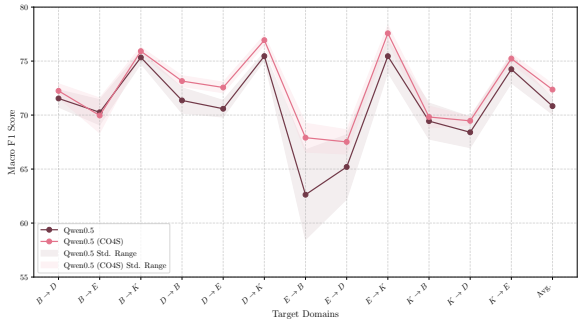


(c) Source and target domains: MNLI.

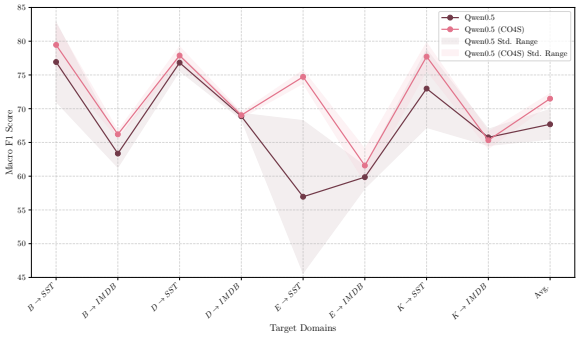


(d) Source domain: MNLI; target domains: SICK and SNLI.

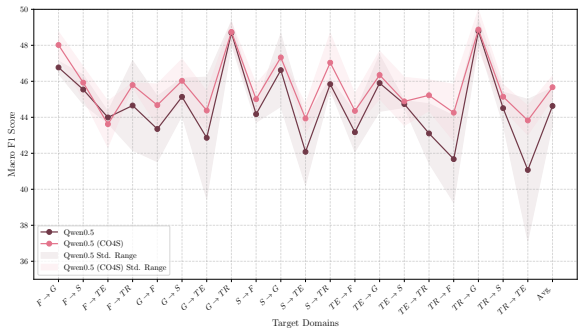
Figure 3: SDG results for RoBERTa-base and RoBERTa-base (CO4S).



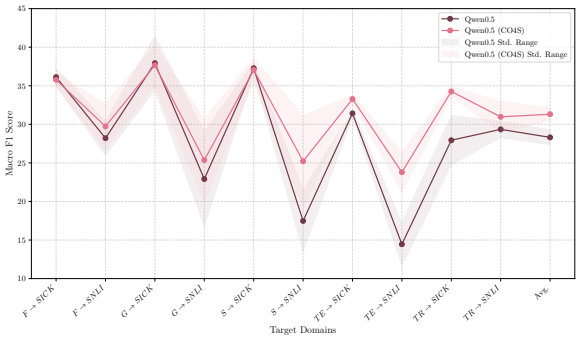
(a) Source and target domains: amazon.



(b) Source domain: amazon; target domains: SST-2 and IMDB.

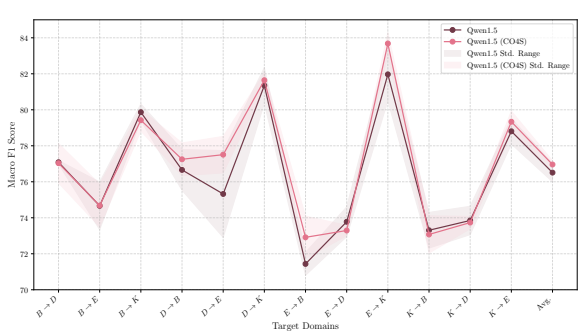


(c) Source and target domains: MNLI.

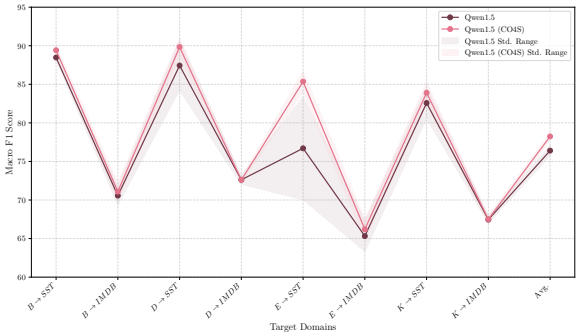


(d) Source domain: MNLI; target domains: SICK and SNLI.

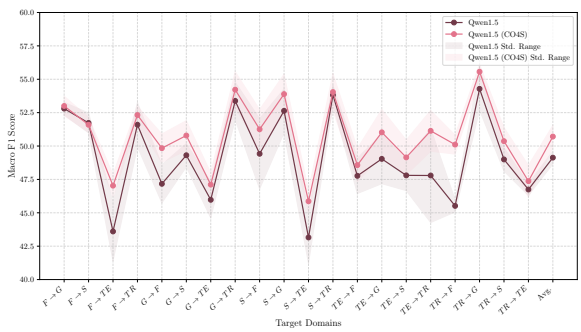
Figure 4: SDG results for Qwen-0.5B and Qwen-0.5B (CO4S).



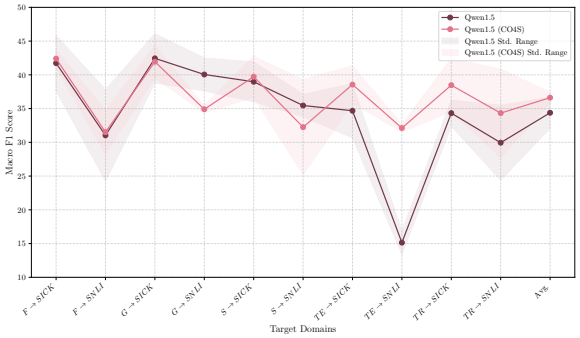
(a) Source and target domains: amazon.



(b) Source domain: amazon; target domains: SST-2 and IMDB.

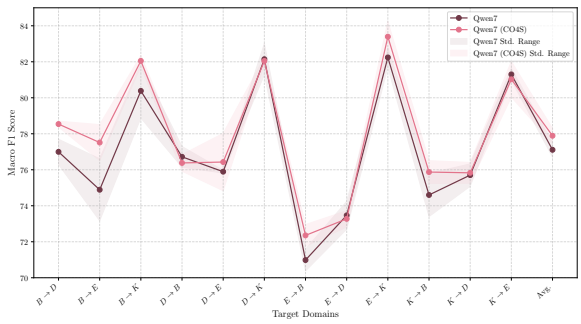


(c) Source and target domains: MNLI.

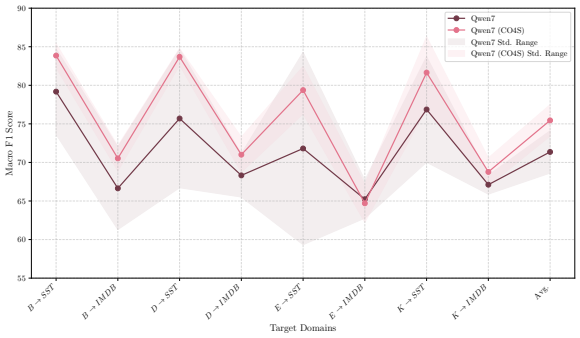


(d) Source domain: MNLI; target domains: SICK and SNLI.

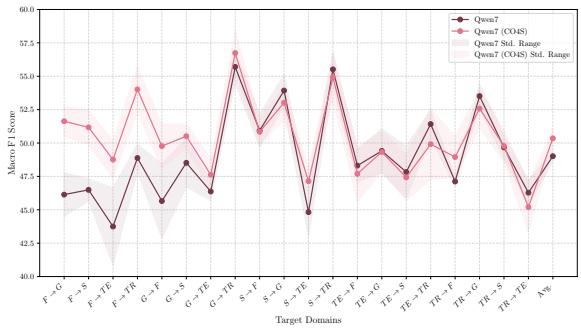
Figure 5: SDG results for Qwen-1.5B and Qwen-1.5B (CO4S).



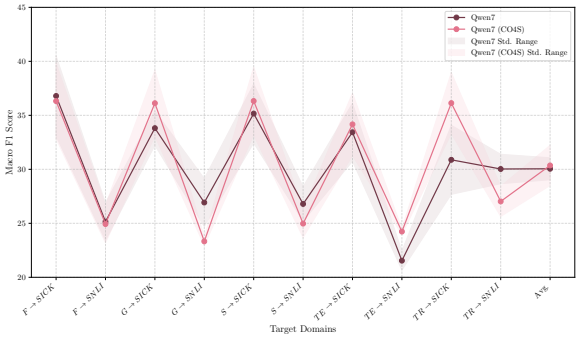
(a) Source and target domains: amazon.



(b) Source domain: amazon; target domains: SST-2 and IMDB.



(c) Source and target domains: MNLI.



(d) Source domain: MNLI; target domains: SICK and SNLI.

Figure 6: SDG results for Qwen-7B and Qwen-7B (CO4S).

Table 10: The LODO results on the SST-2, IMDB, SICK, and SNLI datasets with BERT-base as backbone (%). SA NTP and NLI NTP denote the number of trainable parameters in the SA and NLI tasks, respectively.

Model	SA NTP	NLI NTP	Amazon→SST-2	Amazon→IMDB	MNLI→SICK	MNLI→SNLI	Avg.
PDA	4.1708	6.5363	31.39 ± 6.01	31.70 ± 2.59	EG	EG	-
EAGLE	3.7804	6.1466	42.32 ± 2.25	50.46 ± 4.66	24.53 ± 6.66	17.38 ± 10.28	33.67 ± 13.28
MSCL	0.7890	0.7895	44.19 ± 7.74	75.36 ± 0.76	35.63 ± 1.55	12.73 ± 0.30	41.98 ± 22.44
SwAV	1.2485	1.2493	62.02 ± 5.44	72.17 ± 1.16	35.38 ± 1.85	11.86 ± 9.60	44.79 ± 24.36
ELS	2.3693	4.1449	81.43 ± 0.44	**77.22 ± 0.23**	36.91 ± 5.08	8.55 ± 0.69	51.03 ± 30.06
DomCLP	1.1172	1.1180	77.41 ± 1.78	48.25 ± 18.83	24.17 ± 0.00	13.05 ± 13.16	40.72 ± 24.71
Backbone	**0.0015**	**0.0023**	78.19 ± 2.76	76.09 ± 0.74	34.01 ± 4.43	9.81 ± 1.01	49.53 ± 28.92
CO4S	**0.0015**	**0.0023**	**81.52 ± 0.92**	76.62 ± 0.81	**48.97 ± 2.84**	**23.15 ± 2.24**	**57.57 ± 23.43**
Improve	↓ 99.81%	↓ 99.71%	↑ 0.09	↓ 0.60	↑ 13.34	↑ 5.77	↑ 6.54

Table 11: The LODO results on the Amazon dataset with DistilRoBERTa-base as backbone (%).

Model	NTP	BDE→K	BDK→E	BEK→D	DEK→B	Avg.
PDA	2.4165	79.94 ± 3.01	84.57 ± 0.83	**83.30 ± 0.36**	77.67 ± 4.28	81.37 ± 2.72
EAGLE	2.5969	81.93 ± 3.00	84.06 ± 0.63	79.42 ± 0.62	74.73 ± 5.93	80.04 ± 3.48
MSCL	0.7887	70.12 ± 1.59	66.69 ± 0.34	67.76 ± 1.45	71.87 ± 0.96	69.11 ± 2.02
SwAV	1.2485	72.78 ± 3.25	65.91 ± 9.69	63.46 ± 8.48	63.04 ± 4.25	66.30 ± 3.90
ELS	1.1858	83.25 ± 0.59	84.05 ± 0.05	82.69 ± 0.55	78.17 ± 0.31	82.04 ± 2.28
DomCLP	1.1172	85.28 ± 0.73	84.19 ± 0.33	81.58 ± 0.45	81.93 ± 1.16	83.25 ± 1.54
Backbone	**0.0015**	84.16 ± 1.91	83.69 ± 0.61	82.26 ± 0.18	78.64 ± 2.11	82.19 ± 2.17
CO4S	**0.0015**	**86.41 ± 0.29**	**85.22 ± 0.14**	83.01 ± 0.19	**83.07 ± 0.56**	**84.43 ± 1.45**
Improve	↓ 99.81%	↑ 1.13	↑ 0.65	↓ 0.29	↑ 1.14	↑ 1.18

using electronics as the source domain and kitchen as the target domain, all backbone models exhibit significant performance improvements. Similarly, under the cross-dataset setting with electronics as the source and SST-2 as the target, consistent gains are observed across all backbones. These findings suggest that CO4S is particularly effective in scenarios where the source and target data distributions are relatively balanced.

G Integrating CO4S into other Domain Generalization Models

Since CO4S is designed as a plug-and-play module that can be readily attached to any domain generalization method, this section explores integrating CO4S with existing domain generalization approaches to evaluate whether it can further boost their performance.

Tables 16, 17, and 18 show the LODO results of integrating CO4S with three domain generalization methods (EAGLE, MSCL, and SWAV) using RoBERTa-base as the backbone, evaluated on the Amazon dataset, MNLI dataset, and the cross-dataset setting, respectively.

From these experimental results, we can observe that when the backbone is frozen, CO4S consistently improves the performance of these models, most notably in the cross-dataset setting, where it boosts EAGLE's macro-F1 by 16.63%. However, we also observe that CO4S is not universally beneficial. In particular, its gains on MSCL are marginal, with improvements of only 0.73%, 0.60%, and 1.71% across the three settings, and it even leads to a performance drop of 2.98% on the "book" domain.

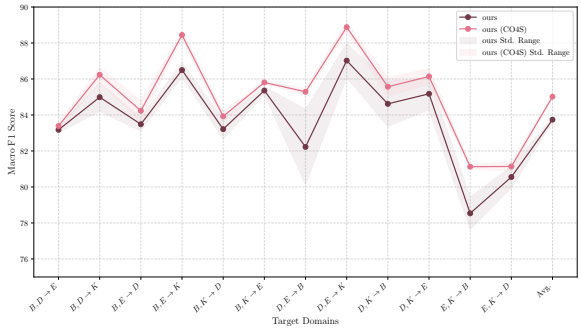
These findings suggest that CO4S can serve as an effective plug-and-play module when integrated with other models, but its success depends on the compatibility of the underlying method. If a model's additional components disrupt or override the PLM's inherent domain-invariant representations, the introduction of CO4S may fail to help or even degrade performance.

H Ablation Study

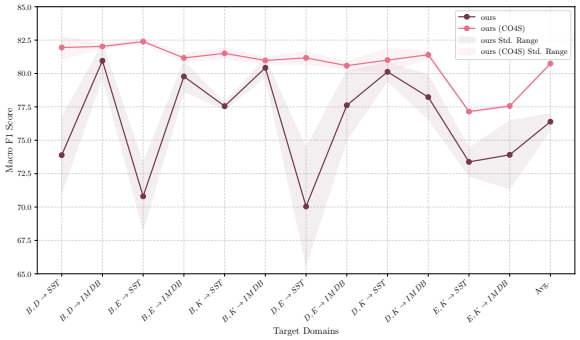
In this section, we provide a detailed explanation for why replacing the mean with the median in our ablation study results in only a slight performance drop. We analyze this phenomenon through the following four evaluation metrics:

- **Relative Euclidean Distance** (denoted as $\mathcal{R}_\mu$): this metric measures the distance between the mean vector and the median vector, which is formulated as:

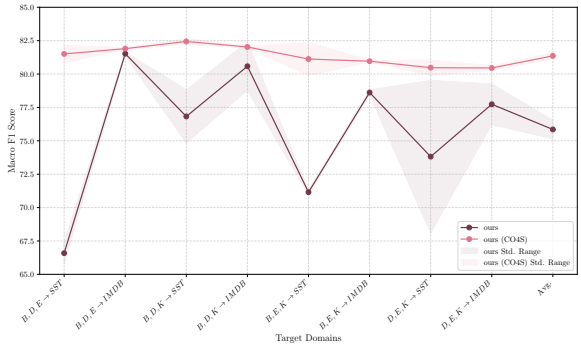
$$\mathcal{R}_\mu = \frac{\|\boldsymbol{c}_{mean} - \boldsymbol{c}_{median}\|_2}{\|\boldsymbol{c}_{mean}\|_2}.$$



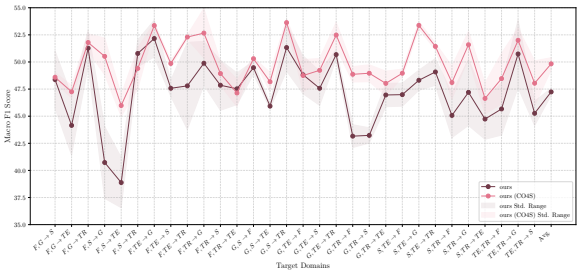
(a) Source and target domains: amazon; number of source domains: 2.



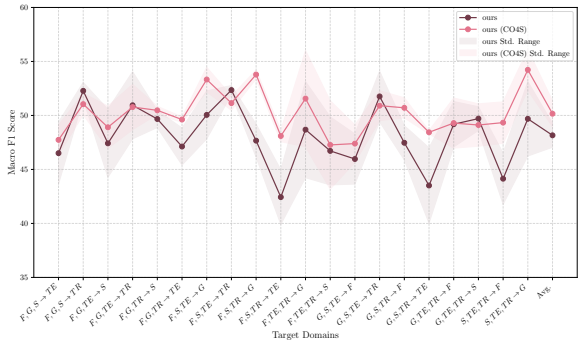
(b) Source domain: amazon; target domains: SST-2 and IMDB; number of source domains: 2.



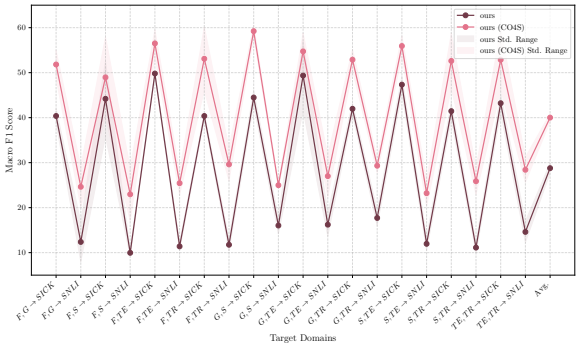
(c) Source domain: amazon; target domains: SST-2 and IMDB; number of source domains: 3.



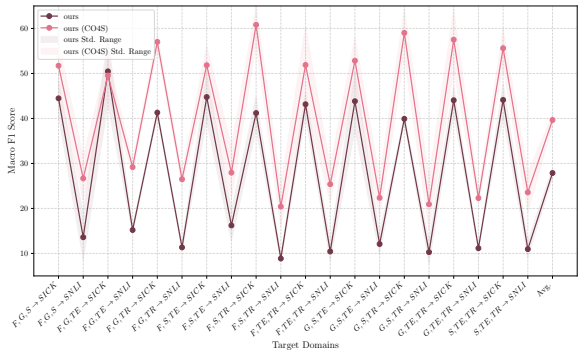
(d) Source and target domains: MNLI; number of source domains: 2.



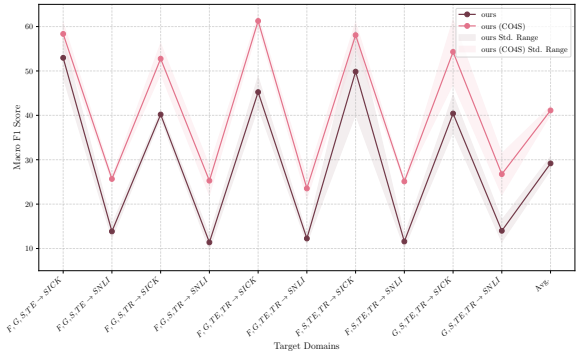
(e) Source and target domains: MNLI; number of source domains: 3.



(f) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 2.

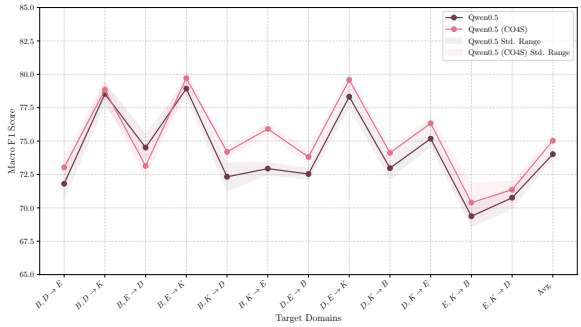


(g) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 3.

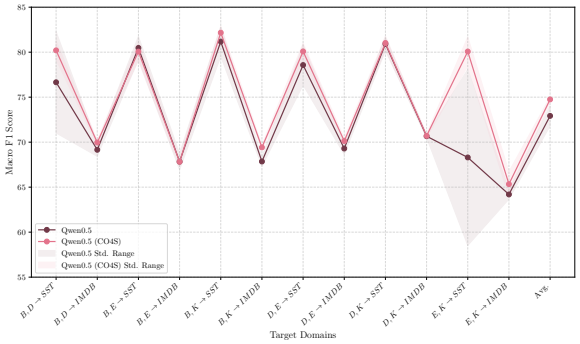


(h) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 4.

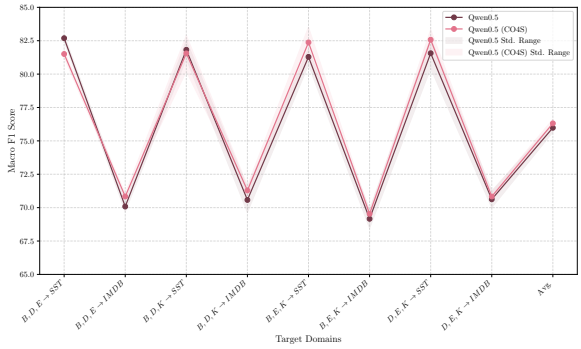
Figure 7: PDG results for RoBERTa-base and RoBERTa-base (CO4S).



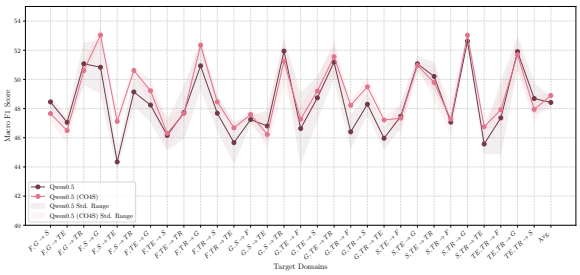
(a) Source and target domains: amazon; number of source domains: 2.



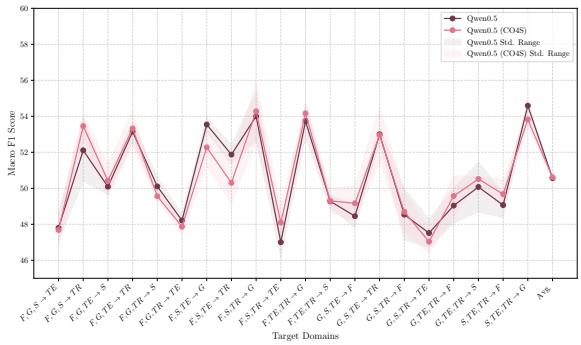
(b) Source domain: amazon; target domains: SST-2 and IMDB; number of source domains: 2.



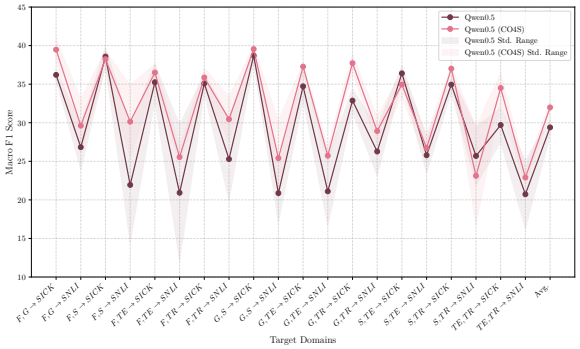
(c) Source domain: amazon; target domains: SST-2 and IMDB; number of source domains: 3.



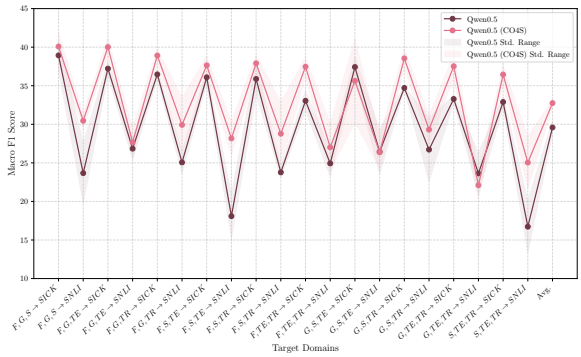
(d) Source and target domains: MNLI; number of source domains: 2.



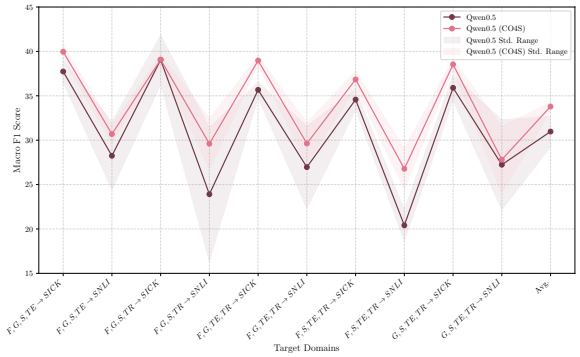
(e) Source and target domains: MNLI; number of source domains: 3.



(f) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 2.

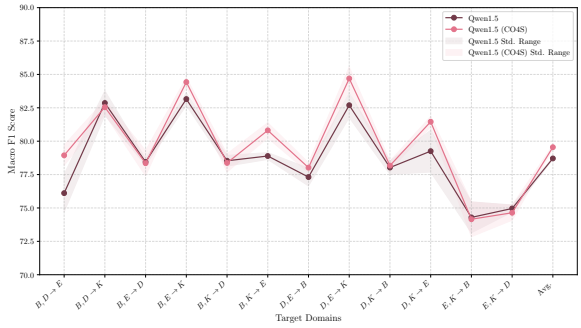


(g) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 3.

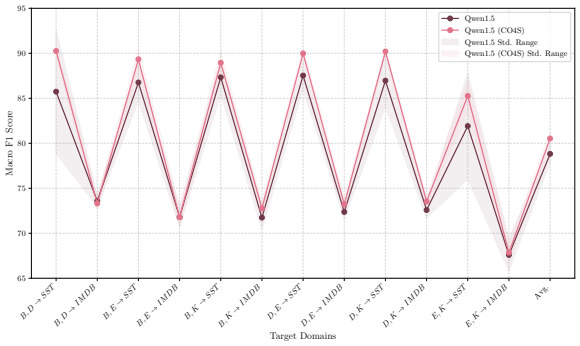


(h) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 4.

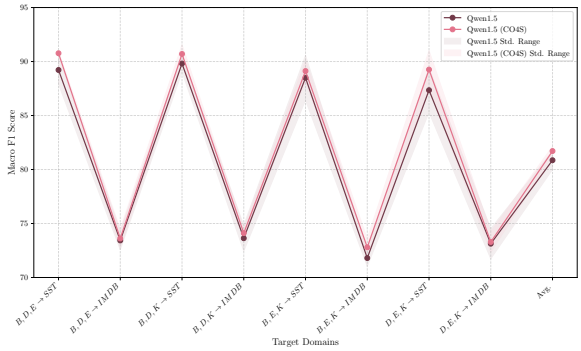
Figure 8: PDG results for Qwen-0.5B and Qwen-0.5B (CO4S).



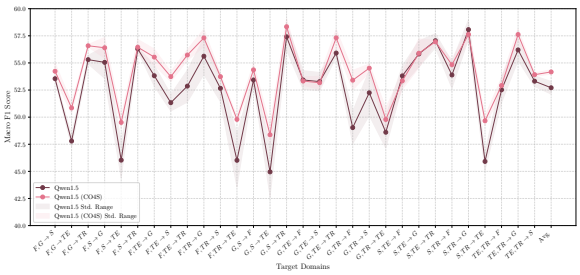
(a) Source and target domains: amazon; number of source domains: 2.



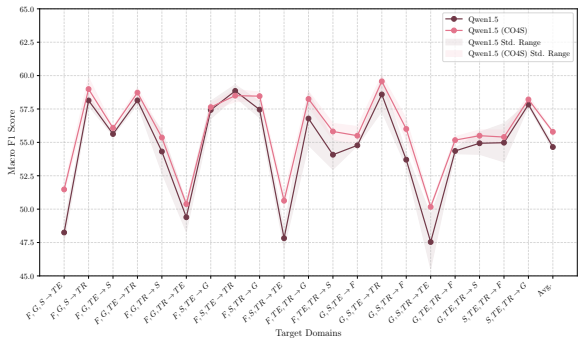
(b) Source domain: amazon; target domains: SST-2 and IMDB; number of source domains: 2.



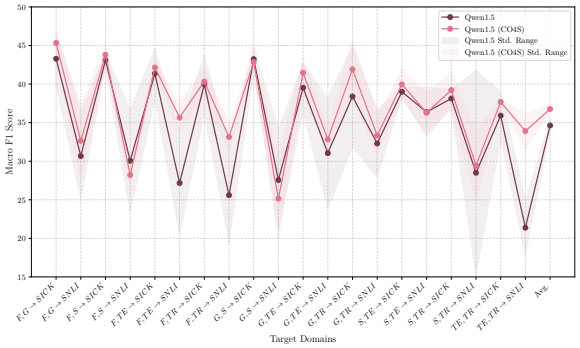
(c) Source domain: amazon; target domains: SST-2 and IMDB; number of source domains: 3.



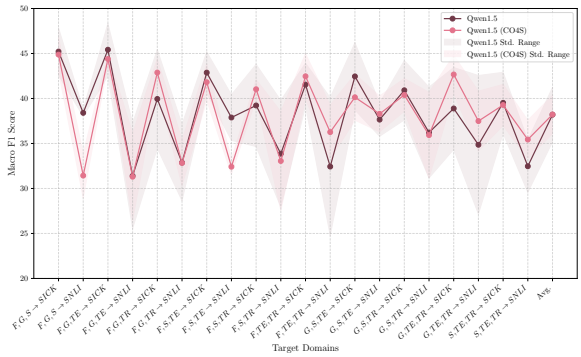
(d) Source and target domains: MNLI; number of source domains: 2.



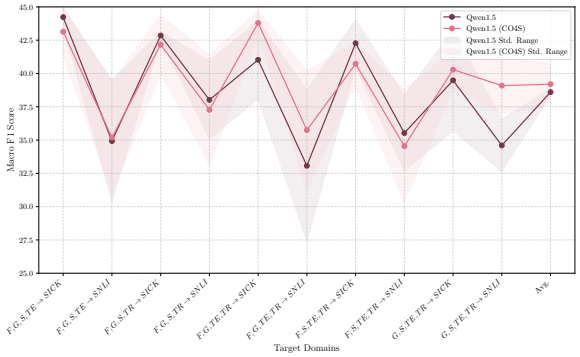
(e) Source and target domains: MNLI; number of source domains: 3.



(f) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 2.

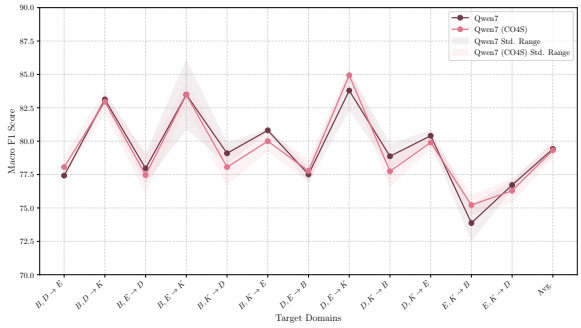


(g) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 3.

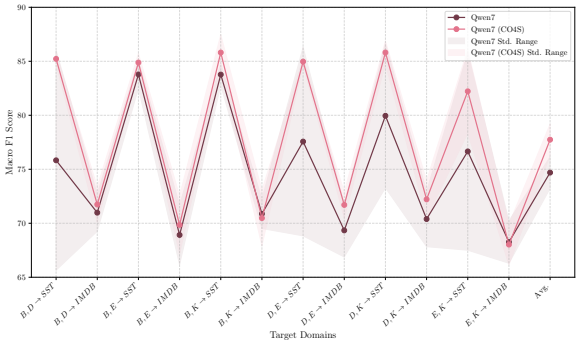


(h) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 4.

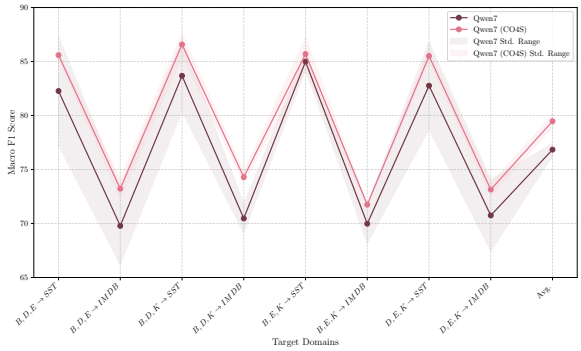
Figure 9: PDG results for Qwen-1.5B and Qwen-1.5B (CO4S).



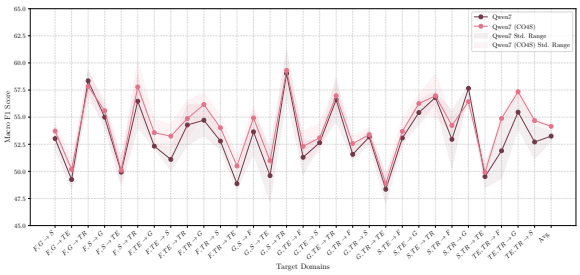
(a) Source and target domains: amazon; number of source domains: 2.



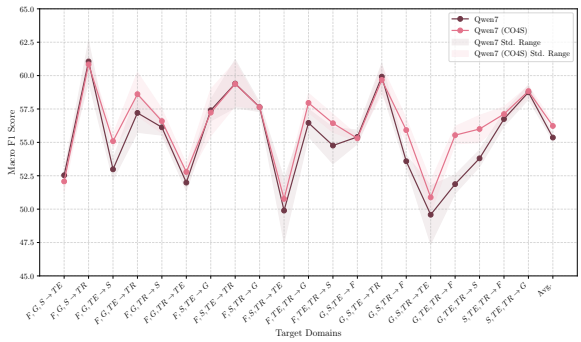
(b) Source domain: amazon; target domains: SST-2 and IMDB; number of source domains: 2.



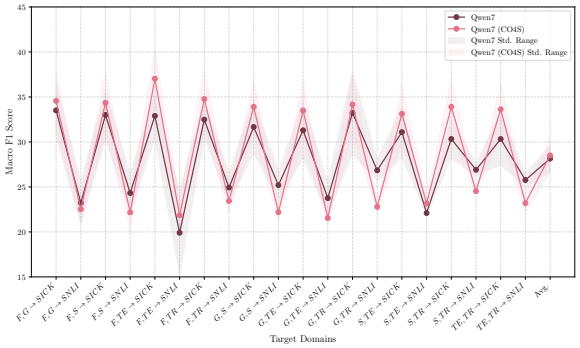
(c) Source domain: amazon; target domains: SST-2 and IMDB; number of source domains: 3.



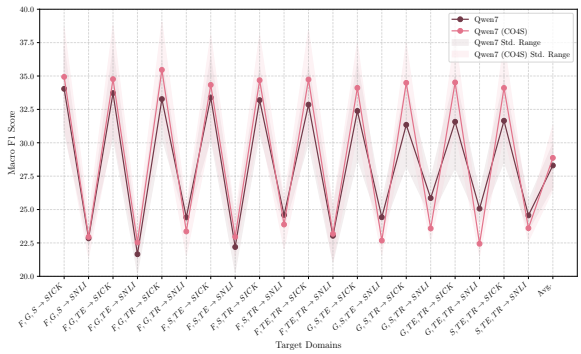
(d) Source and target domains: MNLI; number of source domains: 2.



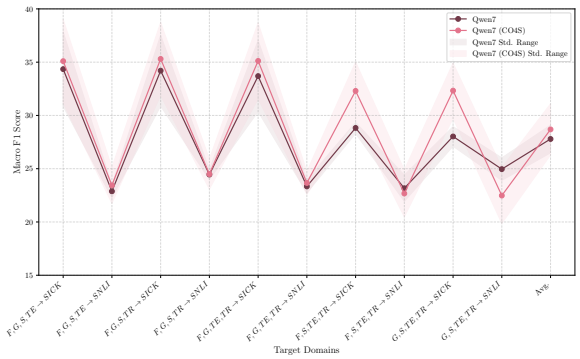
(e) Source and target domains: MNLI; number of source domains: 3.



(f) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 2.



(g) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 3.



(h) Source domain: MNLI; target domains: SICK and SNLI; number of source domains: 4.

Figure 10: PDG results for Qwen-7B and Qwen-7B (CO4S).

Table 12: The LODO results on the MNLI dataset with DistilRoBERTa-base as backbone (%).

Model	NTP	FGST→T*	FGST*→T	FGTT*→G	FSTT*→G	GSTT*→F	Avg.
PDA	4.1906	EG	EG	EG	EG	EG	-
EAGLE	4.3718	36.71 ± 2.42	42.47 ± 1.43	36.14 ± 4.09	38.47 ± 3.26	29.32 ± 2.16	36.62 ± 4.27
MSCL	0.7893	27.80 ± 0.59	30.91 ± 0.20	27.61 ± 0.75	27.02 ± 0.55	32.66 ± 3.19	29.20 ± 2.20
SwAV	1.2493	24.83 ± 1.63	21.67 ± 0.85	26.66 ± 1.72	28.23 ± 1.70	26.13 ± 0.44	25.50 ± 2.20
ELS	2.3701	46.63 ± 1.76	43.82 ± 2.01	45.46 ± 1.73	41.63 ± 1.77	$\mathbf{45.02 \pm 0.58}$	44.51 ± 1.70
DomCLP	1.1180	23.59 ± 5.37	23.77 ± 3.59	19.14 ± 2.65	22.47 ± 2.79	20.27 ± 4.63	21.85 ± 1.84
Backbone	**0.0023**	47.50 ± 1.79	45.24 ± 0.32	46.09 ± 1.41	43.94 ± 2.23	42.33 ± 2.02	45.02 ± 1.78
CO4S	**0.0023**	$\mathbf{48.38 \pm 0.33}$	$\mathbf{46.94 \pm 0.12}$	$\mathbf{46.30 \pm 0.26}$	$\mathbf{48.78 \pm 0.83}$	44.65 ± 0.68	$\mathbf{47.01 \pm 1.49}$
Improve	↓ 99.71%	↑ 0.88	↑ 1.70	↑ 0.21	↑ 4.84	↓ 0.37	↑ 1.99

Table 13: The LODO results on the SST-2, IMDB, SICK, and SNLI datasets with DistilRoBERTa-base as backbone (%). SA NTP and NLI NTP denote the number of trainable parameters in the SA and NLI tasks, respectively.

Model	SA NTP	NLI NTP	Amazon→SST-2	Amazon→IMDB	MNLI→SICK	MNLI→SNLI	Avg.
PDA	4.1906	6.5560	41.08 ± 11.99	36.87 ± 10.79	EG	EG	-
EAGLE	4.3710	6.7372	45.13 ± 6.09	62.97 ± 6.35	17.52 ± 1.86	5.09 ± 2.03	32.68 ± 22.71
MSCL	0.7890	0.7895	49.62 ± 4.42	74.56 ± 2.01	30.61 ± 3.86	13.08 ± 8.06	41.97 ± 22.82
SwAV	1.2485	1.2493	54.28 ± 10.91	65.41 ± 3.81	32.24 ± 1.86	6.41 ± 2.53	39.59 ± 22.57
ELS	2.3693	4.1449	82.58 ± 0.64	81.16 ± 0.20	47.64 ± 7.87	9.47 ± 0.95	55.22 ± 29.88
DomCLP	1.1172	1.1180	83.62 ± 1.13	$\mathbf{83.05 \pm 0.26}$	24.17 ± 0.00	23.87 ± 11.94	53.68 ± 29.66
Backbone	**0.0015**	**0.0023**	81.03 ± 1.36	80.10 ± 1.59	53.43 ± 3.59	11.56 ± 1.40	56.53 ± 28.23
CO4S	**0.0015**	**0.0023**	$\mathbf{84.31 \pm 0.30}$	81.05 ± 0.24	$\mathbf{54.37 \pm 3.10}$	$\mathbf{26.97 \pm 4.93}$	$\mathbf{61.68 \pm 23.16}$
Improve	↓ 99.81%	↓ 99.71%	↑ 0.68	↓ 2.00	↑ 0.94	↑ 3.10	↑ 5.14

- **Skewness Analysis** (denoted as $\mathcal{S}$): this metric assesses whether the sentence embeddings produced by the PLM are symmetric, i.e., whether their mean and median are close, which is formulated as:

$$\text{skew}(Z_j) = \frac{\mathbb{E}\left[(Z_j - \mu_j)^3\right]}{\sigma_j^3},$$

$$\mathcal{S} \coloneqq \frac{1}{d}\sum_{j=1}^{d} \left|\text{skew}(Z_j)\right|,$$

where Z_j is the value in the j-th dimension of the sentence representation $\boldsymbol{h}$; μ_j and σ_j are the mean and standard deviation of the j-th dimension of all sentence representations, respectively; d denotes the total dimensionality of the sentence representations.

- **Cosine Similarity** (denoted as $\mathcal{COS}$): this metric evaluates whether the mean and median vectors share a similar direction, which is formulated as:

$$\mathcal{COS} \coloneqq \cos(\boldsymbol{c}_{mean}, \boldsymbol{c}_{median}).$$

- **Classifier-Aligned Perturbation** (denoted as $\mathcal{A}_c$): this metric determines whether the discrepancy between the mean and median vectors lies primarily in non-discriminative (i.e., task-irrelevant) feature dimensions, which is formulated as:

$$\Delta = \boldsymbol{c}_{mean} - \boldsymbol{c}_{median},$$

$$\mathcal{A}_c \coloneqq \frac{1}{d}\sum_{k=1}^{d} \left|\cos(\Delta, w_c^k)\right|,$$

where w_c^k refers to the k-th dimension of the weights in the classifier f_c; $\cos(a, b)$ computes the cosine similarity between a and b.

Tables 19, 20, and 21 show the experimental results on the Amazon dataset, the MNLI dataset, and cross datasets, respectively.

From these results, we can conclude that:

- Across all datasets, we observe that $\mathcal{R}_\mu$ consistently falls within the range of 2×10^{-3} to 3×10^{-3}, indicating that the mean and median vectors are extremely close in the embedding space. Furthermore, their cosine similarity exceeds 0.99 in all cases, confirming that they share nearly identical directions.

- Skewness analysis reveals that, across all domains, the sentence embeddings produced by the PLM exhibit skewness values around 0.1, suggesting that the distribution of each embedding dimension is highly symmetric and approximately Gaussian. Consequently, the

Table 14: The LODO results on the Amazon dataset with Qwen2.5 as backbone (%).

Model	DEK→B	BEK→D	BDK→E	BDE→K	Avg.
Qwen-0.5B	79.87 ± 0.52	74.97 ± 0.71	74.19 ± 0.23	72.79 ± 1.05	75.45 ± 0.17
Qwen-0.5B (CO4S)	80.36 ± 0.96	75.95 ± 0.62	74.22 ± 0.47	74.54 ± 0.69	76.27 ± 0.39
Improve	↑ 0.49	↑ 0.99	↑ 0.03	↑ 1.75	↑ 0.82
Qwen-1.5B	84.53 ± 1.19	79.48 ± 1.03	78.84 ± 0.58	77.51 ± 0.82	80.09 ± 0.01
Qwen-1.5B (CO4S)	85.10 ± 0.22	81.21 ± 0.55	79.24 ± 0.35	78.52 ± 0.43	81.02 ± 0.28
Improve	↑ 0.58	↑ 1.73	↑ 0.40	↑ 1.00	↑ 0.93
Qwen-7B	85.24 ± 0.35	80.38 ± 0.31	79.04 ± 1.16	78.58 ± 0.66	80.95 ± 0.30
Qwen-7B (CO4S)	85.32 ± 0.49	81.31 ± 0.77	79.91 ± 1.01	78.79 ± 1.37	81.08 ± 0.59
Improve	↑ 0.07	↑ 0.93	↑ 0.57	↑ 0.21	↑ 0.13

Table 15: The LODO results on the MNLI dataset with Qwen2.5 as backbone (%).

Model	FGST→T*	FGST*→T	FGTT*→G	FSTT*→G	GSTT*→F	Avg.
Qwen-0.5B	53.75 ± 0.06	47.86 ± 0.62	50.96 ± 0.96	55.43 ± 0.76	49.29 ± 0.85	51.46 ± 0.30
Qwen-0.5B (CO4S)	54.03 ± 0.16	48.64 ± 0.25	50.99 ± 1.03	55.00 ± 0.60	50.34 ± 0.81	51.80 ± 0.13
Improve	↑ 0.28	↑ 0.78	↑ 0.04	↓ 0.43	↑ 1.05	↑ 0.34
Qwen-1.5B	59.87 ± 0.04	48.90 ± 1.19	56.68 ± 0.54	59.43 ± 0.27	55.50 ± 1.11	56.08 ± 0.26
Qwen-1.5B (CO4S)	61.11 ± 0.43	52.21 ± 0.26	56.79 ± 0.54	59.95 ± 0.65	56.76 ± 0.40	57.07 ± 0.23
Improve	↑ 1.24	↑ 3.31	↑ 0.12	↑ 0.52	↑ 1.26	↑ 0.99
Qwen-7B	60.20 ± 1.27	52.69 ± 0.85	57.65 ± 0.76	59.10 ± 1.19	55.76 ± 1.25	57.08 ± 0.56
Qwen-7B (CO4S)	60.99 ± 1.04	52.71 ± 0.82	58.00 ± 0.20	59.30 ± 0.74	56.92 ± 0.40	57.58 ± 0.22
Improve	↑ 0.79	↑ 0.03	↑ 0.36	↑ 0.20	↑ 1.16	↑ 0.51

mean and median are statistically expected to be very close, i.e., $\boldsymbol{c}_{mean} \approx \boldsymbol{c}_{median}$.

- Finally, $\mathcal{A}_c$ is consistently on the order of 10^{-2} across all domains. This indicates that Δ is nearly orthogonal to the classifier weights, implying that the discrepancy between the mean and median lies primarily in non-discriminative feature dimensions, i.e., in task-irrelevant noise.

Together, these results explain why replacing the mean with the median in CO4S leads to only a minor performance drop: the difference between the two is both small in magnitude and largely confined to features that do not contribute meaningfully to classification.

I Model Efficiency Analysis

In this section, we conduct the model efficiency analysis from 2 aspects: 1) the training time, and 2) the average number of batches processed per second. To ensure a fair comparison, we guarantee that only one method is running on the GPU at any time during the experiments. Specifically, we adopt "Kitchen" as the target domain, fix the batch size to 16, and train all methods for exactly one epoch.

Table 22 shows the experimental results of model efficiency. From these results, we can observe that: 1) Since CO4S is a test-time method, its training time and batches-per-second (BPS) during training are identical to those of the underlying backbone—no additional training overhead is introduced. 2) Our method achieves the fastest training times on both devices, completing training in 20.1465 seconds and 13.4799 seconds, respectively (37.59% and 15.15% faster than the second-fastest method PDA). And 3) Although CO4S and SWAV exhibit similar BPS, CO4S does not require any additional representation learning during training. As a result, CO4S is 210.55% faster in training speed compared to SWAV. These results highlight that CO4S offers exceptional training efficiency, making it highly suitable for resource-constrained scenarios while still achieving strong model generalization and discriminative performance.

J Visualizations

In this section, we visualize the sentence representation distributions using t-SNE (van der Maaten and

Table 16: The LODO experimental results on the Amazon dataset when integrating CO4S with other domain generalization methods (%). We use RoBERTa-base as the backbone.

Model	DEK→B	BEK→D	BDK→E	BDE→K	Avg.
EAGLE	85.01 ± 2.96	82.90 ± 1.92	81.63 ± 1.01	79.59 ± 2.30	82.28 ± 0.57
EAGLE (CO4S)	87.73 ± 0.52	84.64 ± 0.13	82.43 ± 0.59	83.40 ± 0.23	84.55 ± 0.15
Improve	↑ 2.73	↑ 1.74	↑ 0.80	↑ 3.82	↑ 2.27
MSCL	71.98 ± 1.44	66.71 ± 2.03	74.49 ± 1.13	74.62 ± 1.36	71.95 ± 1.45
MSCL (CO4S)	68.99 ± 1.10	71.87 ± 1.95	75.29 ± 1.51	74.57 ± 1.33	72.68 ± 1.44
Improve	↓ 2.98	↑ 5.16	↑ 0.80	↓ 0.05	↑ 0.73
SwAV	65.51 ± 4.77	62.64 ± 4.17	58.12 ± 8.71	63.16 ± 1.12	62.36 ± 3.06
SwAV (CO4S)	64.90 ± 3.73	65.14 ± 5.17	63.29 ± 2.53	64.63 ± 0.61	64.49 ± 2.87
Improve	↓ 0.61	↑ 2.50	↑ 5.17	↑ 1.46	↑ 2.13

Table 17: The LODO experimental results on the MNLI dataset when integrating CO4S with other domain generalization methods (%). We use RoBERTa-base as the backbone.

Model	FGST→T*	FGST*→T	FGTT*→G	FSTT*→G	GSTT*→F	Avg.
EAGLE	38.59 ± 6.10	39.15 ± 3.36	45.38 ± 0.52	39.56 ± 2.32	46.66 ± 3.39	41.87 ± 1.78
EAGLE (CO4S)	41.57 ± 4.41	45.79 ± 2.08	42.77 ± 3.32	45.90 ± 4.38	48.98 ± 1.58	45.00 ± 1.76
Improve	↑ 2.99	↑ 6.64	↓ 2.61	↑ 6.34	↑ 2.32	↑ 3.14
MSCL	31.81 ± 1.77	34.25 ± 0.28	32.89 ± 0.84	33.92 ± 1.16	34.79 ± 2.55	33.53 ± 0.78
MSCL (CO4S)	32.55 ± 2.05	34.87 ± 0.09	32.77 ± 0.99	35.13 ± 1.58	35.35 ± 1.46	34.14 ± 0.95
Improve	↑ 0.74	↑ 0.63	↓ 0.12	↑ 1.21	↑ 0.56	↑ 0.60
SwAV	20.66 ± 2.96	18.77 ± 2.49	23.31 ± 0.45	19.94 ± 14.11	14.97 ± 11.45	19.53 ± 1.21
SwAV (CO4S)	23.19 ± 4.63	18.87 ± 2.93	24.20 ± 2.01	26.17 ± 6.27	24.58 ± 5.31	23.40 ± 3.70
Improve	↑ 2.53	↑ 0.09	↑ 0.89	↑ 6.23	↑ 9.61	↑ 3.87

Hinton, 2008) for both the original model and the model with CO4S. Fig. 11 presents RoBERTa-base results with kitchen and SST-2 as target domains. The visualizations show that applying CO4S introduces negligible changes to the representation space, indicating that our method preserves the domain-invariant features already learned by the pretrained language model, unlike many existing domain generalization approaches that tend to alter them.

To further investigate why the performance improvement of the Qwen models is limited and in some cases even degraded, we visualize the sentence representations obtained from Qwen-0.5B, Qwen-1.5B, and Qwen-7B, as shown in Fig. 12. Compared to the sentence embeddings produced by RoBERTa-base (Fig. 11), we can observe that the sentence representations generated by Qwen appear less compact and exhibit the presence of outliers. These outliers can distort the computed difference vector $\boldsymbol{d}$ between domains, thereby reducing the effectiveness of the generalization and potentially leading to performance degradation.

K Failure Case Study

In this section, we investigate the limitations of CO4S by examining scenarios in which it fails. Specifically, we use the PHEME dataset (Kochkina et al., 2018b)[10] as a testbed. The task associated with this dataset is rumor detection. Notably, this dataset is a low-resource dataset with highly imbalanced domain distributions. There are 5 domains in the PHEME dataset: CharlieHebdo (CH), Ferguson (F), GermanWings (GW), OttawaShooting (OS), and SydneySiege (S). Detailed statistics of the dataset are provided in Table 23, and the distribution is visualized in Fig. 13. From the distribution of the dataset, we observe that the GW domain exhibits relatively uniform data distribution, while the remaining four domains display highly scattered and imbalanced distributions.

K.1 LODO Results on the PHEME Dataset

We utilize RoBERTa-base as the backbone and conduct LODO experiments on the PHEME dataset under both settings: training the backbone (non-freeze) and freezing it (freeze). The experimental results are shown in Table 24. As shown in the

[10]https://github.com/kochkinaelena/Multitask4Veracity

Table 18: The LODO experimental results on the SST-2, IMDB, SICK, and SNLI datasets when integrating CO4S with other domain generalization methods (%). We use RoBERTa-base as the backbone.

Model	Amazon→SST-2	Amazon→IMDB	MNLI→SICK	MNLI→SNLI	Avg.
EAGLE	35.26 ± 1.62	54.84 ± 13.71	20.64 ± 4.44	6.18 ± 3.13	29.23 ± 101
EAGLE (CO4S)	58.59 ± 5.81	64.77 ± 5.67	33.96 ± 6.80	6.10 ± 3.04	61.68 ± 3.09
Improve	$\uparrow$ 23.33	$\uparrow$ 9.93	$\uparrow$ 13.33	$\downarrow$ 0.08	$\uparrow$ 16.63
MSCL	43.66 ± 5.64	73.47 ± 2.53	35.03 ± 1.63	17.24 ± 4.34	26.13 ± 2.67
MSCL (CO4S)	59.29 ± 0.29	75.29 ± 0.03	35.97 ± 1.08	19.71 ± 0.94	27.84 ± 0.96
Improve	$\uparrow$ 15.63	$\uparrow$ 1.82	$\uparrow$ 0.95	$\uparrow$ 2.47	$\uparrow$ 1.71
SwAV	46.87 ± 10.54	60.14 ± 2.93	18.07 ± 2.29	7.59 ± 0.41	33.17 ± 21.20
SwAV (CO4S)	59.50 ± 3.33	62.11 ± 3.86	23.26 ± 5.96	7.71 ± 0.59	38.15 ± 23.33
Improve	$\uparrow$ 12.63	$\uparrow$ 1.98	$\uparrow$ 5.20	$\uparrow$ 0.12	$\uparrow$ 4.98

Table 19: Experimental results in the ablation study comparing the mean and median vectors in terms of relative Euclidean distance, distribution discrepancy, cosine similarity, and weight correlation on the Amazon dataset.

Domain	Metrics	DEK→B	BEK→D	BDK→E	BDE→K
Source	$\mathcal{R}_\mu$	1.99×10^{-3}	1.99×10^{-3}	2.51×10^{-3}	2.45×10^{-3}
	$\mathcal{S}$	0.1003	0.1075	0.1253	0.1255
	$\mathcal{COS}$	> 0.99	> 0.99	> 0.99	> 0.99
	$\mathcal{A}_c$	1.27×10^{-2}	2.22×10^{-2}	3.37×10^{-2}	1.74×10^{-2}
Target	$\mathcal{R}_\mu$	1.68×10^{-3}	2.25×10^{-3}	1.84×10^{-3}	1.95×10^{-3}
	$\mathcal{S}$	0.1069	0.1064	0.1051	0.1060
	$\mathcal{COS}$	> 0.99	> 0.99	> 0.99	> 0.99
	$\mathcal{A}_c$	6.07×10^{-2}	3.82×10^{-2}	5.10×10^{-2}	3.53×10^{-2}

experimental results, when using GW as the target domain, the model performance degrades significantly. Specifically, in the freeze setting, performance drops by 29.99%, while in the non-freeze setting, it drops by 27.43%. Consequently, the average performance across domains decreases by 2.52% and 5.87%, respectively. Furthermore, we observe that fine-tuning the model leads to performance degradation on three target domains: OS ($\downarrow$ 3.96%), GW ($\downarrow$ 27.43%), and CH ($\downarrow$ 2.80%). This suggests that under highly imbalanced domain distributions, training the model will cause overfitting to the source domain, thereby hindering its ability to generalize to unseen domains.

K.2 SDG and PDG results on the PHEME dataset

Given the extensive number of experiments conducted under the SDG and PDG settings, we report only the average performance of the two training strategies (freeze and non-freeze) in Fig. 14 for clarity. Additionally, we highlight cases where CO4S (on freeze setting) exhibits a performance drop exceeding 10% under either setting, as summarized in Table 25. From the experimental results, we observe that under the setting where the backbone model is trained, the average performance of CO4S degrades under both the SDG and PDG training strategies, showing decreases of 4.36%, 4.20%, and 4.03% on three, two, and one source domains, respectively. Notably, the most severe performance drop occurs when GW is used as the target domain in PDG and LODO settings. As illustrated in Figure 13, this can be attributed to the relatively balanced yet small-scale distribution of GW, contrasted with the highly imbalanced and larger-scale distributions of the other four domains. However, under the SDG setting, the performance drop when using GW as the target domain is relatively mild (see Table 26), suggesting that CO4S can still be effective in SDG scenarios where the source and target domains differ significantly. This can be attributed to the fact that, in the SDG setup, there is only a single source domain. In contrast, under the PDG and LODO settings, CO4S averages representations from multiple source domains, and when these domains are highly heterogeneous, the resulting average vector may introduce significant bias, thereby degrading performance.

These findings highlight the limitations of CO4S:

Table 20: Experimental results in the ablation study comparing the mean and median vectors in terms of relative Euclidean distance, distribution discrepancy, cosine similarity, and weight correlation on the MNLI dataset.

Domain	Metrics	FGST→T*	FGST*→T	FGTT*→G	FSTT*→G	GSTT*→F
Source	$\mathcal{R}_\mu$	3.03×10^{-3}	1.72×10^{-3}	2.89×10^{-3}	3.28×10^{-3}	2.78×10^{-3}
	$\mathcal{S}$	0.1058	0.090	0.1057	0.1068	0.1072
	$\mathcal{COS}$	> 0.99	> 0.99	> 0.99	> 0.99	> 0.99
	$\mathcal{A}_c$	2.75×10^{-2}	3.24×10^{-2}	3.08×10^{-2}	2.66×10^{-2}	3.38×10^{-2}
Source	$\mathcal{R}_\mu$	2.01×10^{-3}	5.11×10^{-3}	1.69×10^{-3}	1.65×10^{-3}	2.31×10^{-3}
	$\mathcal{S}$	0.1059	0.1096	0.0994	0.1038	0.1214
	$\mathcal{COS}$	> 0.99	> 0.99	> 0.99	> 0.99	> 0.99
	$\mathcal{A}_c$	4.20×10^{-2}	2.75×10^{-2}	2.48×10^{-2}	1.69×10^{-2}	1.34×10^{-2}

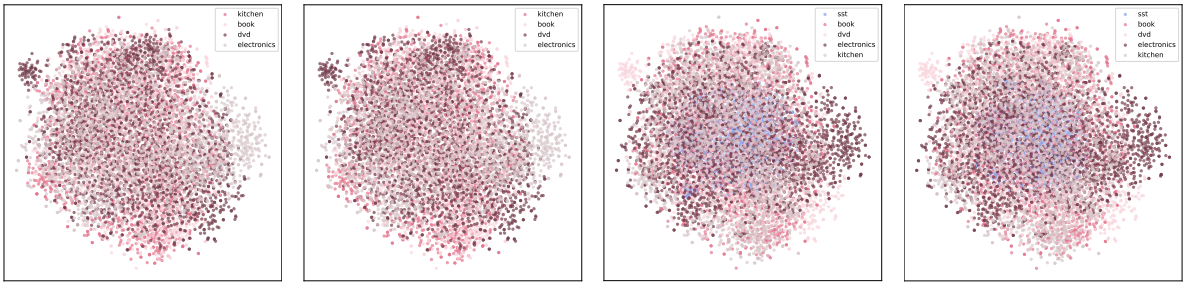
Table 21: Experimental results in the ablation study comparing the mean and median vectors in terms of relative Euclidean distance, distribution discrepancy, cosine similarity, and weight correlation on the SST-2, IMDB, SICK, and SNLI datasets.

Domain	Metrics	Amazon→SST-2	Amazon→IMDB	MNLI→SICK	MNLI→SNLI
Source	$\mathcal{R}_\mu$	2.20×10^{-3}	2.20×10^{-3}	2.72×10^{-3}	2.72×10^{-3}
	$\mathcal{S}$	0.1143	0.1143	0.1032	0.1032
	$\mathcal{COS}$	> 0.99	> 0.99	> 0.99	> 0.99
	$\mathcal{A}_c$	1.87×10^{-2}	7.50×10^{-2}	3.49×10^{-2}	3.49×10^{-2}
Target	1.31×10^{-3}	1.87×10^{-3}	2.45×10^{-3}	5.97×10^{-3}	
	$\mathcal{S}$	0.1060	0.1098	0.2005	0.1741
	$\mathcal{COS}$	> 0.99	> 0.99	> 0.99	> 0.99
	$\mathcal{A}_c$	5.76×10^{-2}	6.57×10^{-2}	2.12×10^{-2}	2.44×10^{-2}

the method is most effective when the domain distributions are relatively balanced and the data volume is sufficiently large. In cases where there are multiple source domains and the distributional shift between the source and target domains is substantial, CO4S becomes less suitable.

Table 22: The models' efficiency analysis. We train each model using the kitchen in the Amazon dataset as the target domain. TT and BPS represent training time and the number of batches processed per second in the training phase, respectively.

Model	V100		H20	
	TT (s) ↓	BPS (s/it) ↑	TT (s) ↓	BPS (s/it) ↑
PDA	32.2805	7.2337	15.8866	14.5256
EAGLE	50.5974	4.5158	32.2699	7.0343
SwAV	62.5652	11.0907	41.4275	**16.7859**
RoBERTa-base	**20.1465**	**11.3838**	**13.4799**	16.6927
RoBERTa-base (CO4S)	**20.1465**	**11.3838**	**13.4799**	16.6927
Qwen-0.5B	47.4415	4.7612	26.4463	8.5165
Qwen-0.5B (CO4S)	47.4415	4.7612	26.4463	8.5165
Qwen-1.5B	168.0943	1.3367	74.4806	3.0179
Qwen-1.5B (CO4S)	168.0943	1.3367	74.4806	3.0179
Qwen-7B	724.7267	0.3101	361.8066	0.6202
Qwen-7B (CO4S)	724.7267	0.3101	361.8066	0.6202



(a) Sentence representation distribution of RoBERTa-base with kitchen as the target domain.

(b) Sentence representation distribution of CO4S with kitchen as the target domain.

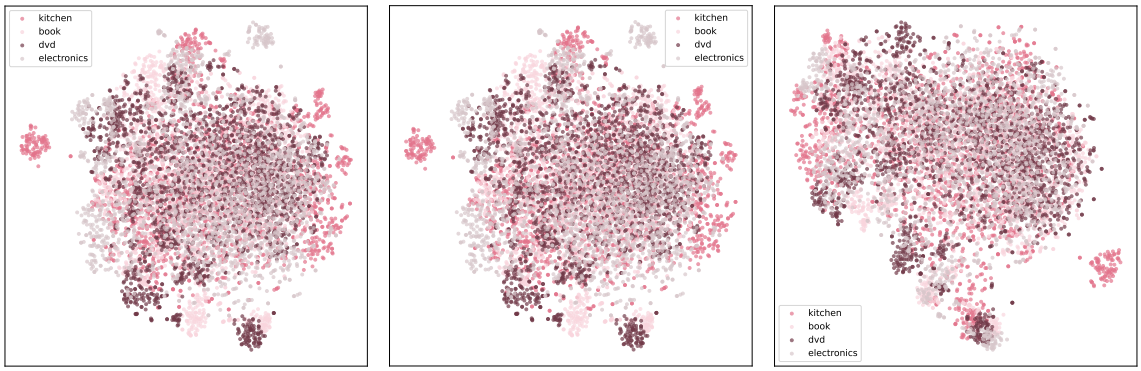
(c) Sentence representation distribution of RoBERTa-base with SST-2 as the target domain.

(d) Sentence representation distribution of CO4S with SST-2 as the target domain.

Figure 11: Sentence representation distribution of RoBERTa-base (w/ and w/o CO4S) with kitchen and SST-2 as the target domain, respectively.

Table 23: The details of the PHEME dataset.

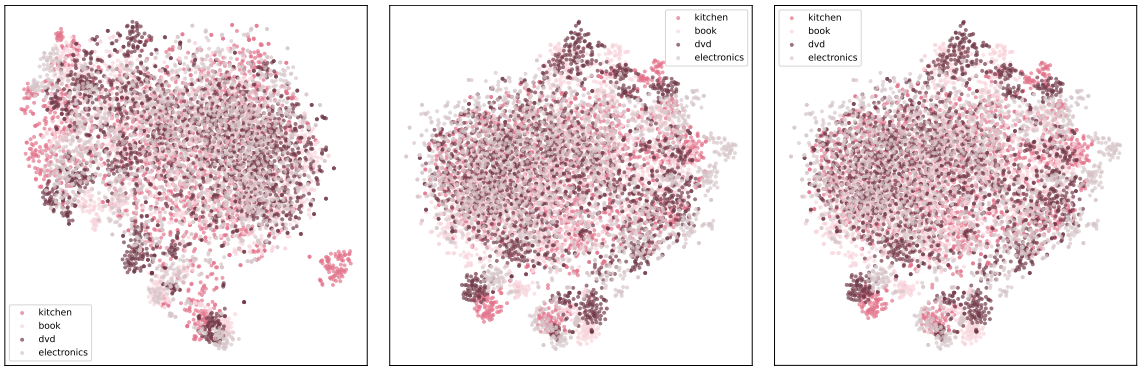
Domains	Train	Test
CH	2,079	2,079
F	1,143	1,143
GW	469	469
OS	890	890
S	1,221	1,221



(a) Sentence representation distribution of Qwen-0.5B with kitchen as the target domain.

(b) Sentence representation distribution of Qwen-0.5B (CO4S) with kitchen as the target domain.

(c) Sentence representation distribution of Qwen-1.5B with kitchen as the target domain.



(d) Sentence representation distribution of Qwen-1.5B (CO4S) with kitchen as the target domain.

(e) Sentence representation distribution of Qwen-7B with kitchen as the target domain.

(f) Sentence representation distribution of Qwen-7B (CO4S) with kitchen as the target domain.

Figure 12: Sentence representation distribution of Qwen-0.5B, Qwen-1.5B, and Qwen-7B (w/ and w/o CO4S) with kitchen as the target domain.

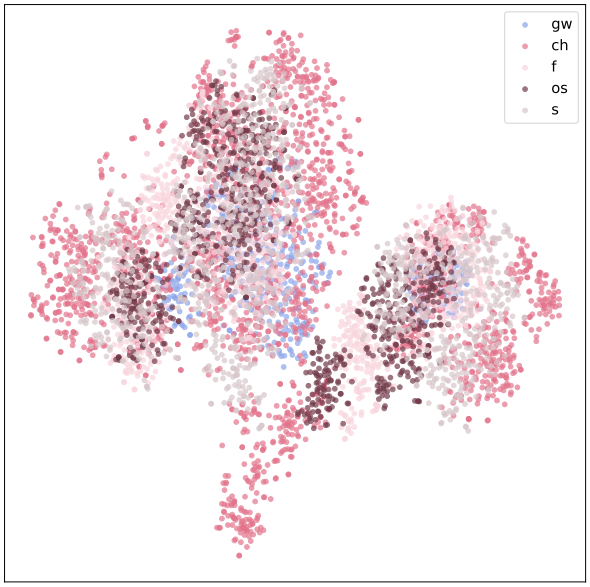


Figure 13: The domain distributions of the PHEME dataset.

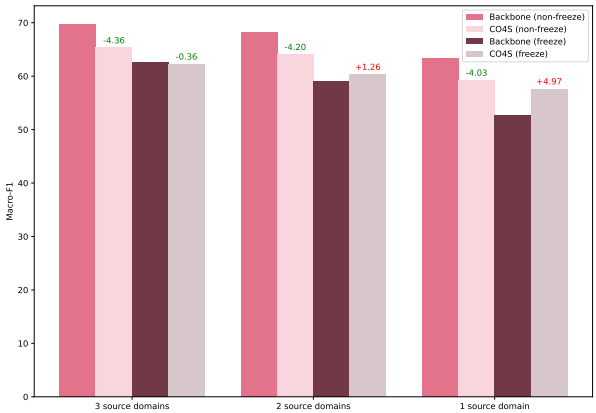


Figure 14: The average performance of RoBERTa-base (freeze and non-freeze) on the PHEME dataset with SDG and PDG settings. The numbers represent the performance increase, with red indicating an improvement and green indicating a decline.

Table 24: The LODO results on the PHEME dataset (%). All experiments are implemented using RoBERTa-base. The column headers represent the target domains. Non-freeze refers to settings where the pretrained model parameters are fine-tuned, while freeze indicates that the pretrained parameters are kept fixed during training.

Model	S	OS	GW	F	CH	Avg.
Backbone (non-freeze)	70.65 ± 2.31	72.24 ± 1.12	68.48 ± 2.89	62.90 ± 2.68	79.17 ± 1.39	70.69 ± 1.58
CO4S (non-freeze)	70.98 ± 1.96	68.28 ± 2.19	41.05 ± 1.38	67.38 ± 0.40	76.38 ± 1.41	64.81 ± 0.66
Improve	$\uparrow 0.33$	$\downarrow 3.96$	$\downarrow 27.43$	$\uparrow 4.49$	$\downarrow 2.80$	$\downarrow 5.87$
Backbone (freeze)	60.76 ± 3.13	57.59 ± 2.65	74.17 ± 0.28	50.76 ± 3.12	75.88 ± 1.47	63.83 ± 1.65
CO4S (freeze)	63.57 ± 3.02	57.68 ± 2.50	44.18 ± 0.78	64.35 ± 1.09	76.79 ± 0.16	61.32 ± 1.21
Improve	$\uparrow 2.81$	$\uparrow 0.09$	$\downarrow 29.99$	$\uparrow 13.59$	$\uparrow 0.91$	$\downarrow 2.52$

Table 25: The PDG results on the PHEME dataset (%). This table reports only the source domains and target domains along with the corresponding decrease in performance, where the performance of CO4S (freeze) decreased by more than 10% under the two settings.

Model	CH, OS $\rightarrow$ GW	CH, S $\rightarrow$ GW	F, OS $\rightarrow$ GW	F, S $\rightarrow$ GW
Backbone (freeze)	65.86 ± 2.78	61.19 ± 7.51	71.48 ± 0.72	65.72 ± 4.33
CO4S (freeze)	49.38 ± 4.67	36.06 ± 2.50	56.38 ± 3.16	46.41 ± 1.45
Decrease	$\downarrow 16.48$	$\downarrow 25.16$	$\downarrow 15.09$	$\downarrow 19.30$
Model	CH, F, OS $\rightarrow$ GW	CH, F, S $\rightarrow$ GW	CH, OS, S $\rightarrow$ GW	F, OS, S $\rightarrow$ GW
Backbone (freeze)	61.64 ± 3.42	62.92 ± 1.57	77.08 ± 1.01	73.12 ± 1.45
CO4S (freeze)	46.24 ± 0.80	39.36 ± 1.87	44.17 ± 3.93	60.22 ± 3.70
Decrease	$\downarrow 15.40$	$\downarrow 23.56$	$\downarrow 32.91$	$\downarrow 12.89$

Table 26: The SDG results on the PHEME dataset (%). This table reports only using GW as the target domain. We use RoBERTa-base under freezing setting as the backbone.

Model	CH $\rightarrow$ GW	F $\rightarrow$ GW	OS $\rightarrow$ GW	S $\rightarrow$ GW
Backbone (freeze)	37.56 ± 0.49	35.30 ± 1.01	70.91 ± 0.95	63.35 ± 2.26
CO4S (freeze)	34.38 ± 1.36	36.77 ± 1.89	71.00 ± 0.84	63.89 ± 5.81
Improve	$\downarrow 3.18$	$\uparrow 1.47$	$\uparrow 0.09$	$\uparrow 0.54$