Balanced Direction from Multifarious Choices: Arithmetic Meta-Learning for Domain Generalization

Supplementary Material

1. Proof of Gradient Matching

This section provides a detailed proof of gradient matching that the gradient of step k is matched with those of the previous k-1 steps.

Preliminary. Let's start by revisiting the definitions of the inner loop of n steps, during which the model's parameters transition from Θ to $\hat{\Theta}$. We represent the loss at each step as $\{\mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_n\}$, and the parameter updating trajectory as $\{\theta_1, \theta_2, ..., \theta_{n+1}\}$, with θ_1 and θ_{n+1} corresponding to Θ and $\hat{\Theta}$ respectively. We use $\mathcal{L}_i(\theta_j)$ to denote the loss of the i-th step on parameters θ_j. During the inner loop, the update process is performed with a small learning rate α:

$$\begin{aligned} \theta_2 &= \theta_1 - \alpha \nabla \mathcal{L}_1(\theta_1) \\ \theta_3 &= \theta_2 - \alpha \nabla \mathcal{L}_2(\theta_2) \\ &\vdots \\ \theta_{n+1} &= \theta_n - \alpha \nabla \mathcal{L}_n(\theta_n). \end{aligned} \tag{1}$$

Objective. To prove that for any $i = k$, step k is gradient-matched with the previous k-1 steps as:

$$\mathcal{L}_k(\theta_k) = \mathcal{L}_k(\theta_1) - \alpha \sum_{i=1}^{k-1} \nabla \mathcal{L}_i(\theta_1) \cdot \nabla \mathcal{L}_k(\theta_1) + \mathcal{O}(\alpha^2), \tag{2}$$

it is adequate to demonstrate that the following equation holds for any loss function $\mathcal{L}$:

$$\mathcal{L}(\theta_k) = \mathcal{L}(\theta_1) - \alpha \sum_{i=1}^{k-1} \nabla \mathcal{L}_i(\theta_1) \cdot \nabla \mathcal{L}(\theta_1) + \mathcal{O}(\alpha^2). \tag{3}$$

Base Case. When i equals 1, it is evident that $\mathcal{L}(\theta_i) = \mathcal{L}(\theta_1)$, so Eq. (3) holds. When i equals 2, we can substitute Eq. (1) into $\mathcal{L}(\theta_2)$ and conduct a first order Taylor expansion on it:

$$\mathcal{L}(\theta_2) = \mathcal{L}(\theta_1) - \alpha \nabla \mathcal{L}_1(\theta_1) \cdot \nabla \mathcal{L}(\theta_1) + \mathcal{O}(\alpha^2), \tag{4}$$

thus Eq. (3) is also valid.

Inductive Step. Given that Eq. (3) is true for arbitrary $i \leq k$, we proceed to establish its validity for the case when i equals $k+1$. Plugging Eq. (1) and Eq. (3) into $\mathcal{L}(\theta_{k+1})$ yields:

$$\begin{aligned} \mathcal{L}(\theta_{k+1}) &= \mathcal{L}(\theta_k) - \alpha \nabla \mathcal{L}_k(\theta_k) \cdot \nabla \mathcal{L}(\theta_k) + \mathcal{O}(\alpha^2) \\ &= \mathcal{L}(\theta_1) - \alpha \sum_{i=1}^{k-1} \nabla \mathcal{L}_i(\theta_1) \cdot \nabla \mathcal{L}(\theta_1) + \mathcal{O}(\alpha^2) \\ &\quad - \alpha(\nabla \mathcal{L}_k(\theta_1) + \mathcal{O}(\alpha))(\nabla \mathcal{L}(\theta_1) + \mathcal{O}(\alpha)) + \mathcal{O}(\alpha^2) \\ &= \mathcal{L}(\theta_1) - \alpha \sum_{i=1}^{k} \nabla \mathcal{L}_i(\theta_1) \cdot \nabla \mathcal{L}(\theta_1) + \mathcal{O}(\alpha^2). \end{aligned} \tag{5}$$

Note that we substitute $\nabla \mathcal{L}_k(\theta_k)$ into Eq. (3) to obtain:

$$\nabla \mathcal{L}_k(\theta_k) = \nabla \mathcal{L}_k(\theta_1) - \alpha \sum_{i=1}^{k-1} \nabla \mathcal{L}_i(\theta_1) \mathcal{H}_k(\theta_1) + \mathcal{O}(\alpha^2). \tag{6}$$

$\mathcal{H}_k(\theta_1)$ is a Hessian left-multiplied by $\nabla \mathcal{L}_i(\theta_1)$. Eq. (6) is simplified as $\nabla \mathcal{L}_k(\theta_1) + \mathcal{O}(\alpha)$ in Eq. (5), and $\nabla \mathcal{L}(\theta_k)$ follows the same process.

Conclusion. We have shown that Eq. (3) is valid for all $i = k$ and for any loss function $\mathcal{L}$. Therefore, our objective of Eq. (2) is successfully demonstrated.

2. Other Results.

We illustrate detailed results of Arith, as shown in Tab. 1. We also provide results from five datasets within the multi-modal WILDS benchmark [5] as mentioned in main text. AMAZON, CAMELYON17 [1], CIVILCOMMENTS [3], IWILDCAM [2], and FMOW [4] present diverse challenges across multiple domains and modalities, and we adopt the hyperparameter configuration from [6] to ensure consistency and comparability in our experiments.

- AMAZON comprises 1.4 million customer reviews from 7,676 customers, with the goal of predicting a score (1-5 stars) for each review.
- CAMELYON17 consists of 450,000 lymph node scans from five hospitals for cancer detection.
- CIVILCOMMENTS includes 450,000 comments collected from online articles, each annotated for toxicity and mentions of demographic identities.
- IWILDCAM contains over 200,000 wildlife photos captured by stationary cameras across 324 locations, aimed at identifying 186 species.
- FMOW features satellite images from five regions over a span of 16 years, encompassing 62 categories.

3. Other Analysis.

Why a balanced positioning? The good balance refers to updating the model towards the centroid of domain experts, which integrates model averaging but differs in some key aspects (Sec. 2.5). This averaging can be viewed as a parameter-efficient form of ensemble learning, with a single model estimating the ensemble output of multiple domain experts. Consider a update trajectory $\{\theta_1, \theta_2, ..., \theta_n\}$, where $\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n}\theta_i$ and $f(\cdot)$ is the model's output. The Taylor expansion of the output ensemble is:

$$\frac{1}{n}\sum_{i=1}^{n} f(\theta_i) = f(\hat{\theta}) + \frac{1}{n}\sum_{i=1}^{n}(\theta_i - \hat{\theta})^T \nabla f(\hat{\theta}) + \mathcal{O}(\alpha). \quad (7)$$

The second term equals 0 because $\sum_{i=1}^{n}(\theta_i - \hat{\theta}) = 0$, and the third term is $\mathcal{O}(||\max_{i=1}^{n}(\theta_i - \hat{\theta})||^2)$. Along the same update trajectory, the different domain-optimal parameters are relatively close to each other, resulting in a smaller $(\theta_i - \hat{\theta})$, thus $f(\hat{\theta}) \approx \frac{1}{n}\sum_{i=1}^{n} f(\theta_i)$, indicating that our method closely estimates the ensemble output of domain experts.

Discussion about computation and memory cost. Our computation and memory costs are similar to other meta-learning methods. The computation cost primarily arises from backpropagation, which occurs only in the inner loop that we do not modify, thus keeping this cost comparable to other methods. Although increasing the number of steps raises costs, all comparisons are conducted with the same number of steps. For example, the training time for Fish and our method with 5000 iterations on the PACS dataset is 85.4 *min* and 91.9 *min*, respectively. The main memory cost is due to the computation graph generated by backpropagation. Our method continuously accumulates the gradients during the inner loop to update parameters without generating additional computation graphs. As a result, the extra memory overhead is limited to storing these gradients, which is no larger than the size of the model's inherent parameters. For example, it is 90M for ResNet50, which is negligible compared to the total cost of approximately 6000M with a batch size of 32 for three domains.

References

[1] Peter Bandi, Oscar Geessink, Quirine Manson, Marcory Van Dijk, Maschenka Balkenhol, Meyke Hermsen, Babak Ehteshami Bejnordi, Byungjae Lee, Kyunghyun Paeng, Aoxiao Zhong, et al. From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge. *IEEE Transactions on Medical Imaging*, 2018. 1

[2] Sara Beery, Elijah Cole, and Arvi Gjoka. The iwildcam 2020 competition dataset. *arXiv preprint arXiv:2004.10340*, 2020. 1

[3] Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion Proceedings of The 2019 World Wide Web Conference*, 2019. 1

[4] Gordon Christie, Neil Fendley, James Wilson, and Ryan Mukherjee. Functional map of the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1

[5] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International conference on Machine Learning*, pages 5637–5664. PMLR, 2021. 1

[6] Yuge Shi, Jeffrey Seely, Philip Torr, N Siddharth, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. In *International Conference on Learning Representations*, 2021. 1

Table 1. Detailed results on DomainBed benchmark.

Domain Index	PACS	VLCS	OfficeH	TerraInc	DomainNet
Domain 1	85.9 $\pm$ 0.5	98.7 $\pm$ 0.3	64.6 $\pm$ 0.8	52.3 $\pm$ 1.9	59.0 $\pm$ 0.4
Domain 2	81.3 $\pm$ 1.0	64.3 $\pm$ 0.8	55.3 $\pm$ 0.9	42.4 $\pm$ 2.5	19.7 $\pm$ 0.2
Domain 3	97.1 $\pm$ 0.5	76.0 $\pm$ 0.9	78.3 $\pm$ 0.4	57.5 $\pm$ 1.3	47.0 $\pm$ 0.3
Domain 4	81.8 $\pm$ 1.0	78.6 $\pm$ 1.0	79.4 $\pm$ 0.6	40.2 $\pm$ 2.3	12.7 $\pm$ 0.3
Domain 5	-	-	-	-	59.4 $\pm$ 0.4
Domain 6	-	-	-	-	51.1 $\pm$ 0.7
Avg	86.5 $\pm$ 0.3	79.4 $\pm$ 0.3	69.4 $\pm$ 0.1	48.1 $\pm$ 1.2	41.5 $\pm$ 0.1

Table 2. Results on AMAZON (%)

Method	Average acc	10th acc	Worst acc
ERM	70.3	50.7	4.0
Fish	70.6	51.1	5.3
Arith	**70.7**	**52.0**	**5.3**

Table 3. Accuracy on CAMELYON17 (%)

Method	20	21	22	23	24	25	26	27	28	29	Avg
ERM	49.2	30.2	73.6	74.8	64.4	60.8	57.0	37.8	89.6	77.3	73.1
Fish	52.4	**36.0**	72.3	**77.5**	69.0	65.1	59.3	**43.6**	90.0	77.6	74.8
Arith	**54.4**	33.8	**83.6**	75.2	**72.5**	**69.5**	**64.0**	40.7	**90.1**	**79.9**	**76.6**

Table 4. Accuracy on CIVILCOMMENTS (%)

Method	N1	N2	N3	N4	N5	N6	N7	N8	T1	T2	T3	T4	T5	T6	T7	T8	Avg
ERM	82.8	84.4	72.0	89.9	77.8	83.8	70.5	71.2	**82.7**	**82.8**	**78.7**	**79.0**	**77.1**	76.1	80.7	80.5	87.4
Fish	84.9	86.4	76.8	90.6	**80.9**	**85.5**	72.5	**73.9**	79.8	79.9	73.2	76.5	74.1	76.3	80.1	79.6	87.9
Arith	**87.8**	**89.3**	**77.9**	**91.5**	80.6	85.4	**73.5**	72.1	77.3	76.1	73.2	75.0	74.8	**77.1**	**80.9**	**81.1**	**90.0**

Table 5. Results on IWILDCAM (%)

Method	Average acc	Recall macro	F1 macro
ERM	61.6	23.4	20.7
Fish	62.2	22.7	21.1
Arith	**63.2**	**25.2**	**22.5**

Table 6. Accuracy on FMOW (%)

Method	2016	2017	Asi	Eur	Afr	Ame	Oce	Avg
ERM	53.4	47.0	51.9	54.8	33.3	54.4	58.7	51.6
Fish	53.7	47.5	52.7	**55.0**	33.9	**54.6**	**59.0**	52.0
Arith	**53.8**	**47.9**	**54.5**	54.8	**34.1**	54.2	57.2	**52.2**