Supplementary Material of "Understanding Hessian Alignment for Domain Generalization"

Sobhan Hemati* Guojun Zhang* Amir Estiri Xi Chen
Huawei Noah's Ark Lab
{sobhan.hemati, guojun.zhang, amir.hossein.estiri1, xi.chen4}@huawei.com

A. Proof of Proposition 4

Proof. To formulate the classifier head of the neural network, let z_i be the i-th component of the feature vector before the classifier layer and the classifier's parameter $\boldsymbol{\theta}$ is decomposed to $w_{k,i}$, the element in row k and column i of the classifier weight matrix, and b_k, the bias term for the k-th output. We define a_k as

$$a_k = \sum_{i}^{c} w_{k,i} z_i + b_k, \tag{24}$$

where c is the number of classes. Given a_k, if we assume the classifier activation function $\sigma(\cdot)$ to be softmax, the classifier output for the k-th neuron can be written as

$$\hat{y}_k = \sigma(a_k) = \frac{e^{a_k}}{\sum_{j=1}^{c} e^{a_j}}, \tag{25}$$

Now, if we denote $(\boldsymbol{x}, \boldsymbol{y})$ as the sample and $\hat{\boldsymbol{y}}$ as the associated output, and assume the loss function be cross-entropy

$$\ell(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\theta}) = -\sum_{l=1}^{c} y_l \log \hat{y}_l, \tag{26}$$

Having the loss function, we proceed to calculate the gradients and hessian of loss with respect to classifier parameters $w_{p,q}$ and b_p is

$$\frac{\partial \ell(\hat{\boldsymbol{y}}, \boldsymbol{y}; \boldsymbol{\theta})}{\partial w_{p,q}} = \sum_{l=1}^{c} \frac{\partial \ell}{\partial \hat{y}_l} \sum_{k=1}^{c} \frac{\partial \hat{y}_l}{\partial a_k} \frac{\partial a_k}{\partial w_{p,q}}. \tag{27}$$

$$\frac{\partial \ell(\hat{\boldsymbol{y}}, \boldsymbol{y}; \boldsymbol{\theta})}{\partial b_u} = \sum_{l=1}^{c} \frac{\partial \ell}{\partial \hat{y}_l} \sum_{k=1}^{c} \frac{\partial \hat{y}_l}{\partial a_k} \frac{\partial a_k}{\partial b_u}. \tag{28}$$

Given that the activation function is a softmax function $\hat{y}_l = \sigma(a_l) = \frac{e^{a_l}}{\sum_j e^{a_j}}$, the $\frac{\partial \sigma(a_l)}{\partial a_k}$ can be calculated as:

$$\frac{\partial \sigma(a_l)}{\partial a_k} = \sigma(a_l)\delta_{l,k} - \sigma(a_l)\sigma(a_k). \tag{29}$$

Now, using Eq. 29, we can rewrite Eqs. 27 and 28 as follows:

$$\frac{\partial \ell}{\partial w_{p,q}} = (\hat{y}_p - y_p) z_q, \ \frac{\partial \ell}{\partial b_u} = (\hat{y}_u - y_u). \tag{30}$$

*Equal contribution.

For the Hessian, we only calculate the elements of matrix that are only related to the classifier layer. More precisely, we calculate $\frac{\partial^2 \ell}{\partial w_{u,v} \partial w_{p,q}}$, $\frac{\partial^2 \ell}{\partial w_{p,q} \partial b_u}$, and $\frac{\partial^2 \ell}{\partial b_u \partial b_v}$. For the $\frac{\partial^2 \ell}{\partial w_{u,v} \partial w_{p,q}}$ we can write

$$\frac{\partial^2 \ell}{\partial w_{u,v} \partial w_{p,q}} = \frac{\partial}{\partial w_{u,v}} \left((\hat{y}_p - y_p) z_q \right). \tag{31}$$

To calculate the above expression, we need $\frac{\partial \hat{y}_p}{\partial w_{u,v}}$:

$$\frac{\partial \hat{y}_p}{\partial w_{u,v}} = \sum_k \frac{\partial \hat{y}_p}{\partial a_k} \frac{\partial a_k}{\partial w_{u,v}} = \sum_k \frac{\partial \sigma(a_p)}{\partial a_k} \sum_i \delta_{k,u} \delta_{i,v} z_i = \sum_k \sigma(a_p)(\delta_{p,k} - \sigma(a_k)) \delta_{k,u} z_v = \hat{y}_p z_v (\delta_{p,u} - \hat{y}_u) \tag{32}$$

Now, incorporating eq. 32 into eq. 31, the elements of Hessian matrix in classifier layer i.e., $\frac{\partial^2 \ell}{\partial w_{u,v} \partial w_{p,q}}$, we have:

$$\frac{\partial^2 \ell}{\partial w_{u,v} \partial w_{p,q}} = z_q z_v \hat{y}_p (\delta_{p,u} - \hat{y}_u). \tag{33}$$

For $\frac{\partial^2 \ell}{\partial w_{p,q} \partial b_u}$ we can write

$$\frac{\partial^2 \ell}{\partial w_{p,q} \partial b_u} = \frac{\partial}{\partial b_u} \left((\hat{y}_p - y_p) z_q \right) = \frac{\partial (\hat{y}_p - y_p)}{\partial b_u} z_q = \sum_k \frac{\partial \hat{y}_p}{\partial a_k} \frac{\partial a_k}{\partial b_u} z_q = z_q \hat{y}_p (\delta_{p,u} - \hat{y}_u). \tag{34}$$

Eventually, for $\frac{\partial^2 \mathcal{L}}{\partial b_u \partial b_v}$ we have:

$$\frac{\partial^2 \ell}{\partial b_u \partial b_v} = \frac{\partial}{\partial b_v} \left(\hat{y}_u - y_u \right) = \sum_k \frac{\partial \hat{y}_u}{\partial a_k} \frac{\partial a_k}{\partial b_v} = \hat{y}_u (\delta_{u,v} - \hat{y}_v). \tag{35}$$

□

B. Alignment attributes in Hessian and gradient in regression task with mean square error loss and general activation function

In this section, we extend our analysis to regression tasks for real numbers. We show that the classifier head's gradient and Hessian yield similar information of the features. To adapt our framework to regression, we replace the meaning of σ from softmax to an arbitrary uni-variate activation function.

Proposition 5 (Alignment attributes in Hessian and gradient for mean square error loss and general activation function). *Let $\hat{y}$ and y be the network prediction and true target associated with the output neuron of a single output network, $\sigma(\cdot)$ be the activation function, z_i be the i-th feature value before the last layer (regression layer). Suppose the last layer's parameter $\boldsymbol{\theta}$ is decomposed to w_i, the i-th element of the weight vector, and b, the bias term. Matching the gradients and Hessians with respect to the last layer across the domain aligns the following attributes*

$$\frac{\partial \ell}{\partial b} = (\hat{y} - y)\sigma'(a), \tag{36}$$

$$\frac{\partial \ell}{\partial w_i} = (\hat{y} - y)\sigma'(a) z_i, \tag{37}$$

$$\frac{\partial^2 \ell}{\partial b^2} = \sigma'(a)^2 + (\hat{y} - y)\sigma''(a), \tag{38}$$

$$\frac{\partial^2 \ell}{\partial w_i \partial b} = \sigma'(a)^2 z_i + (\hat{y} - y)\sigma''(a) z_i, \tag{39}$$

$$\frac{\partial^2 \ell}{\partial w_i \partial w_k} = \sigma'(a)^2 z_i z_k + (\hat{y} - y)\sigma''(a) z_i z_k, \tag{40}$$

Proof. To formulate the last layer of the neural network we define a as

$$a = \sum_i w_i z_i + b. \tag{41}$$

Given a, if we assume the last layer activation function is $\sigma(\cdot)$, the single output can be written as

$$\hat{y} = \sigma(a). \tag{42}$$

Now, if we denote the input data as $(\boldsymbol{x}, y)$ and associated output $\hat{y}$, and assume the loss function be

$$\ell(\boldsymbol{x}, y; \boldsymbol{\theta}) = \frac{1}{2}(\hat{y} - y)^2. \tag{43}$$

Having the loss function, we proceed to calculate the gradients and hessian of loss with respect to last layer parameters w_i and b. For the gradients, we write

$$\frac{\partial \ell}{\partial w_i} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a} \frac{\partial a}{\partial w_i} = (\hat{y} - y)\sigma'(a) z_i, \tag{44}$$

$$\frac{\partial \ell}{\partial b} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial a} \frac{\partial a}{\partial b} = (\hat{y} - y)\sigma'(a). \tag{45}$$

For the Hessian matrix, we only calculate the elements of the matrix that are only related to the last layer. More precisely, we calculate $\frac{\partial^2 \mathcal{L}}{\partial w_i \partial w_k}$, $\frac{\partial^2 \mathcal{L}}{\partial w_i \partial b}$, and $\frac{\partial^2 \mathcal{L}}{\partial b^2}$. For the $\frac{\partial^2 \mathcal{L}}{\partial w_i \partial w_k}$ we can write

$$\frac{\partial^2 \ell}{\partial w_i \partial w_k} = \frac{\partial}{\partial w_k}\left((\hat{y} - y) \cdot \sigma'(a) z_i\right) = \frac{\partial (\hat{y} - y)}{\partial w_k} \cdot \sigma'(a) z_i + (\hat{y} - y) \frac{\partial}{\partial w_k} \sigma'(a) z_i \tag{46}$$

To calculate the above expression, we need $\frac{\partial \hat{y}}{\partial w_k}$ and $\frac{\partial}{\partial w_k}\sigma'(a)$.

$$\frac{\partial \hat{y}}{\partial w_k} = \frac{\partial \hat{y}}{\partial a} \frac{\partial a}{\partial w_k} = \sigma'(a) z_k, \tag{47}$$

$$\frac{\partial}{\partial w_k}\sigma'(a) = \frac{\partial}{\partial a}\sigma'(a) \frac{\partial a}{\partial w_k} = \sigma''(a) z_k. \tag{48}$$

Now, incorporating eq. 48 into 46 the elements of the last-layer Hessian matrix become:

$$\frac{\partial^2 \ell}{\partial w_i \partial w_k} = \sigma'(a)^2 z_i z_k + (\hat{y} - y)\sigma''(a) z_i z_k. \tag{49}$$

For $\frac{\partial^2 \mathcal{L}}{\partial w_i \partial b}$ we can write

$$\frac{\partial^2 \ell}{\partial w_i \partial b} = \frac{\partial}{\partial b}\left((\hat{y} - y) \cdot \sigma'(a) z_i\right) = \frac{\partial (\hat{y} - y)}{\partial b} \cdot \sigma'(a) z_i + (\hat{y} - y) \frac{\partial}{\partial b} \sigma'(a) z_i \tag{50}$$

$$= \frac{\partial \hat{y}}{\partial a} \frac{\partial a}{\partial b} \cdot \sigma'(a) z_i + (\hat{y} - y) \frac{\partial}{\partial a}\sigma'(a) \frac{\partial a}{\partial b} z_i = \sigma'(a)^2 z_i + (\hat{y} - y)\sigma''(a) z_i. \tag{51}$$

Eventually, $\frac{\partial^2 \mathcal{L}}{\partial b^2}$ is calculated as:

$$\frac{\partial^2 \ell}{\partial b^2} = \frac{\partial}{\partial b}\left((\hat{y} - y) \cdot \sigma'(a)\right) = \frac{\partial (\hat{y} - y)}{\partial b} \cdot \sigma'(a) + (\hat{y} - y) \frac{\partial}{\partial b} \sigma'(a) \tag{52}$$

$$= \frac{\partial \hat{y}}{\partial a} \frac{\partial a}{\partial b} \cdot \sigma'(a) + (\hat{y} - y) \frac{\partial}{\partial a}\sigma'(a) \frac{\partial a}{\partial b} = \sigma'(a)^2 + (\hat{y} - y)\sigma''(a). \tag{53}$$

□

Eqs. 45, 49, 51, 53 show that matching Hessians and gradients with respect to the last layer parameters will match neural network outputs, last layer input features and covariance between output features across domains. This supports the idea of matching gradients and Hessians during training. The above result can be extended to multi-dimensional outputs if the activation is element-wise.

C. Ablation Study

Table 7: Comparison of ERM, IRM, V-Rex, Fishr, and our proposed methods HGP and Hutchinson with ablation study on α and β on Colored MNIST. The setting is same as the Colored MNIST experiment introduced in IRM (Arjovsky et al., 2019).

Method	Train acc.	Test acc.
ERM	86.4 ± 0.2	14.0 ± 0.7
IRM	71.0 ± 0.5	65.6 ± 1.8
V-REx	71.7 ± 1.5	67.2 ± 1.5
Fishr	71.0 ± 0.9	69.5 ± 1.0
HGP	71.0 ± 1.5	69.4 ± 1.3
HGP ($\alpha = 0$)	70.6 ± 1.8	69.3 ± 1.2
HGP ($\beta = 0$)	78.9 ± 0.3	53.3 ± 1.7
Hutchinson	61.7 ± 1.9	**74.0 ± 1.2**
Hutchinson ($\alpha = 0$)	70.6 ± 1.8	69.3 ± 1.2
Hutchinson ($\beta = 0$)	84.9 ± 0.1	9.8 ± 0.2

Table 8: Comparison of ERM, IRM, V-Rex, Fishr, and our proposed methods HGP and Hutchinson with ablation study on α and β on imbalanced Colored MNIST where each domain has 95% from one class and 5% from other class. Except for the imposed label shift, the setting is same as the Colored MNIST experiment introduced in IRM (Arjovsky et al., 2019).

Method	Train acc.	Test acc.
ERM	86.4 ± 0.1	16.7 ± 0.1
IRM	84.9 ± 0.1	14.3 ± 1.4
V-REx	83.3 ± 0.2	35.1 ± 1.2
Fishr	75.7 ± 2.7	35.5 ± 5.3
HGP	83.0 ± 0.2	30.0 ± 0.9
HGP ($\alpha = 0$)	83.2 ± 0.4	29.7 ± 1.7
HGP ($\beta = 0$)	84.3 ± 0.1	20.4 ± 1.0
Hutchinson	79.4 ± 0.3	**47.7 ± 1.4**
Hutchinson ($\alpha = 0$)	83.2 ± 0.4	29.7 ± 1.7
Hutchinson ($\beta = 0$)	84.9 ± 0.1	9.8 ± 0.2

We also repeat the Colored MNIST and imbalanced Colored MNIST experiments in scenarios where one of α and β is non-zero, in Table 7 and Table 8. Recall that α controls the Hessian alignment and β controls the gradient alignment. If not mentioned, the values for α and/or β are non-zero and they are chosen exactly as the IRM paper (Arjovsky et al., 2019). According to Table 7, for both HGP and Hutchinson methods, gradient alignment seems to contribute more to OOD generalization on CMNIST. This might be due to the heavy correlation shift and we have to align the local minima first. For Hutchinson, when $\beta = 0$, the OOD performance drops which we believe is because the value that has been chosen for α is optimized for the IRM loss. In other words, if we optimize α for aligning the diagonal part of Hessian, it can contribute to the OOD generalization. For imbalanced Colored MNIST, the same trend for the role of α and β can be observed. Overall, the key observation is that both aligning gradients and diagonal parts of Hessians contribute to the OOD generalization.

D. Domainbed Results for Other Model Section Methods

In this section, we provide the Domainbed results for the two other model selection methods, i.e., the training-domain validation set and test-domain validation set (oracle). First note that the oracle model selection is not a valid benchmarking scheme and not applicable in practice as it uses the target domain data for selecting the hyperparameters. In fact, in this scenario, algorithms with more hyperparameters and training tricks (like warmup, exponential moving average and etc) can obtain better performance since they have more freedom to tune the model on test data. Considering this, we should not rely on the oracle model selection technique to compare domain generalization algorithms. The other model selection technique is

the training-domain validation set where the validation sets of all training domain are concatenated together and select the hyperparameters that maximize the accuracy on the entire validation set.

As can be seen in Table 9 and Table 10, although Hessian alignment methods are not the best, their performance across all datasets is still competitive for other model selections. As also shown in Gulrajani and Lopez-Paz (2020), different model selections could result in different rankings of the algorithms. We find that training-domain model selection in general gives better results for most baseline algorithms, but the performance of the Hutchinson method slightly degrades. We defer the study of comparing model selections to future work.

Table 9: DomainBed benchmark with *training-domain validation set model selection* method for CMNIST, VLCS, PACS, and OfficeHome datasets. We show the best and second best number with boldface and underline respectively.

Algorithm	**VLCS**	**PACS**	**OfficeHome**	**DomainNet**	**Avg**
ERM	77.5	85.5	66.5	40.9	67.6
IRM	78.5	83.5	64.3	33.9	65.1
GroupDRO	76.7	84.4	66.0	33.3	65.1
Mixup	77.4	84.6	68.1	39.2	67.3
MLDG	77.2	84.9	66.8	41.2	67.5
CORAL	**78.8**	<u>86.2</u>	**68.7**	41.5	**68.8**
MMD	77.5	84.6	66.3	23.4	63.0
DANN	<u>78.6</u>	83.6	65.9	38.3	66.6
CDANN	77.5	82.6	65.8	38.3	66.1
MTL	77.2	84.6	66.4	40.6	67.2
SagNet	77.8	**86.3**	68.1	40.3	68.1
ARM	77.6	85.1	64.8	35.5	65.8
VREx	78.3	84.9	66.4	33.6	65.8
RSC	77.1	85.2	65.5	38.9	66.7
AND-mask	78.1	84.4	65.5	37.2	66.3
SAND-mask	77.4	84.6	65.8	32.1	65.0
Fish	77.8	85.5	<u>68.6</u>	**42.7**	<u>68.7</u>
Fishr	77.8	85.8	67.8	<u>41.7</u>	68.3
HGP	77.6	84.7	68.2	41.1	67.9
Hutchinson	76.8	83.9	68.2	41.6	67.6

Table 10: DomainBed benchmark with *test-domain validation set (oracle)* model selection method for CMNIST, VLCS, PACS, and OfficeHome datasets. We show the best and second best number with boldface and underline respectively.

Algorithm	**VLCS**	**PACS**	**OfficeHome**	**DomainNet**	**Avg**
ERM	77.6	86.7	66.4	41.3	68.0
IRM	76.9	84.5	63.0	28.0	63.1
GroupDRO	77.4	<u>87.1</u>	66.2	33.4	66.0
Mixup	78.1	86.8	68.0	39.6	68.1
MLDG	77.5	86.8	66.6	41.6	68.1
CORAL	77.7	<u>87.1</u>	**68.4**	41.8	<u>68.8</u>
MMD	77.9	**87.2**	66.2	23.5	63.7
DANN	<u>79.7</u>	85.2	65.3	38.3	67.1
CDANN	**79.9**	85.8	65.3	38.5	67.4
MTL	77.7	86.7	66.5	40.8	67.9
SagNet	77.6	86.4	67.5	40.8	68.1
ARM	77.8	85.8	64.8	36.0	66.1
VREx	78.1	**87.2**	65.7	30.1	65.3
RSC	77.8	86.2	66.5	38.9	67.4
AND-mask	76.4	86.4	66.1	37.9	66.7
SAND-mask	76.2	85.9	65.9	32.2	65.1
Fish	77.8	85.8	66.0	<u>42.7</u>	68.1
Fishr	78.2	86.9	<u>68.2</u>	**43.4**	**69.2**
HGP	77.3	86.5	67.4	41.2	68.1
Hutchinson	77.9	86.3	**68.4**	41.9	68.6