APPENDIX

We structure the appendix as follows: We present additional experiments in Section A and we describe the details of the experiments in Section B. In Section C we provide proofs for the propositions in the paper and additional discussion on the methods.

A ADDITIONAL EXPERIMENTS

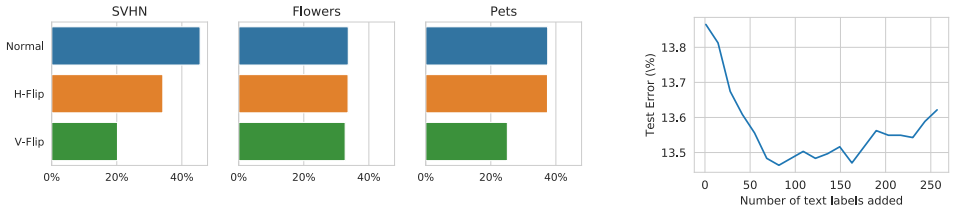

Figure 5: **(Left) DIVA can select the best data augmentation for each task.** We optimize the probability with which each data augment transformation is applied. DIVA optimizes the transformation to apply for the particular task. **(Right) Use of DIVA extend on a multi-modal task.** Selecting only the most beneficial text embeddings to use in a multi-modal classification task (as scored by DIVA) outperforms blindly using all available text embeddings.

Validation overfitting. When updating the dataset using the dataset derivative there is a risk of overfitting to the validation set after repeated applications of the derivative. Namely the validation loss is initially an unbiased estimate of the test loss yet after using it to update the dataset repeatedly it eventually will start overfitting. In our settings, we notice that when using a small number of gradient updates (< 5) and with step sizes $\eta \sim 0.1$ we are able to avoid overfitting and improve the *test* error when optimizing the dataset derivative based on the validation loss. In this experiment we present the final test and validation classification errors of optimized datasets. As we optimize with respect to the validation loss, it is indeed clear that the validation loss decreases dramatically yet more importantly are the effect of the test accuracy.

Dataset	DIVA Extend	Uniform	Improvement
Aircrafts	58.00	60.01	+2.01
Cub-200	39.42	42.29	+2.87
MIT Indoor-67	32.54	33.73	+1.19
Oxford Flowers	20.56	23.29	+2.73
Stanford Cars	60.37	62.45	+2.09
Caltech-256	21.97	24.55	+2.59

Table 2: Results of using DIVA Extend to select the best samples to extend several fine-grain classification datasets. We train a linear classifier on top of a ResNet-34 pretrained on ImageNet, and compare the test performance when extending the target training dataset with 50% of the pool samples selected either uniformly at random or via DIVA Extend.

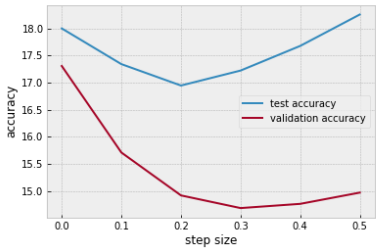

Figure 6: The test error decreases as the dataset is optimized with respect to the validation set until eventually overfitting commences. The validation set error decreases more significantly as the dataset is optimized directly on the validation set, yet for very large step-sizes the first order optimization becomes inaccurate. The plot uses the Caltech-256 dataset to illustrate the overfitting

DIVA Extend plots. In Figure 4 we report the results on all the remaining datasets following the set-up of Figure 4. In Table 2 we report the accuracy of DIVA Extend and uniform sampling on the various datasets when adding 50% of samples from the pool.

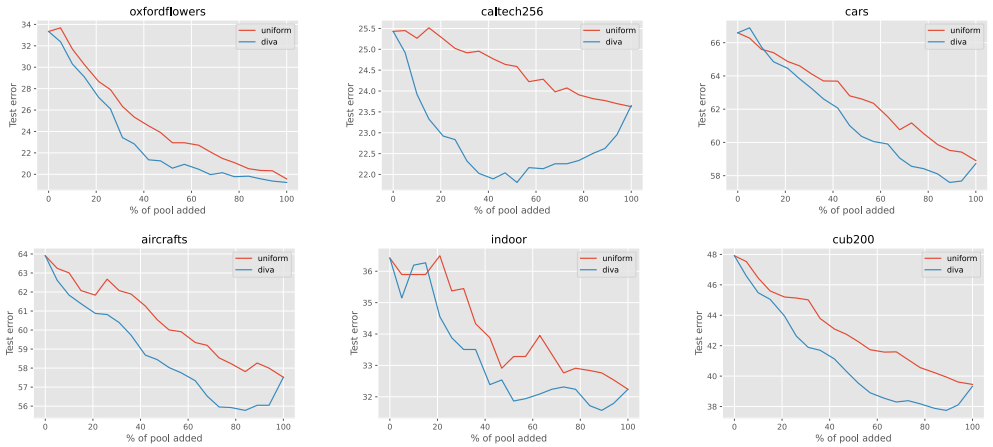

Figure 7: Same plots as Figure 4 on the other fine-grained datasets.

B EXPERIMENTAL DETAILS

Dataset details For our experiments we utilize several fine-grain classification datasets from the computer vision community that are standard for fine-tuning image classification tasks (CUB-200 Welinder et al. (2010), FGVC-Aircraft, Maji et al. (2013), Stanford Cars Krause et al. (2013), Caltech-256 Griffin et al. (2007), Oxford Flowers 102 Nilsback & Zisserman (2008), MIT-67 Indoor Quattoni & Torralba (2009), Street View House Number Netzer et al. (2011), and the Oxford Pets Parkhi et al. (2012)). Some of the datasets do not follow a default train-test split and we use the following splits commonly used in the literature for the datasets,

- Oxford Flowers 102: We use the original 1020 images in the training split without incorporating the default validation set.
- Caltech-256: We split the dataset into a training set with 60 images from each class for training, and use remaining data for testing.

Pre-training setup For the pre-trained networks we use for fine-tuning, we use the pre-trained configurations available on PyTorch's torchvision package. In particular the reference networks are pre-trained using the ImageNet Deng et al. (2009) dataset. The images embedded by the network are pre-processed via standard resizing and center cropping (256 resize, followed by a 224 cropping).

Regularization parameter λ: To get the best unweighted dataset baseline to compare with the optimized dataset, for each of the un-optimized original datasets, we first search for optimal λ values in $\lambda \in \{2^n \text{ for } n \in \{-20, -19, \dots 4\}\}$ to measure the classifier's performance. After selecting optimal λ we proceed with dataset optimization with the optimal λ values. Note that DIVA does not require λ to be optimal and improvements are even more significant for un-optimized λ.

Dataset Derivative Computation In Section C we derive the closed-form dataset derivatives used for DIVA. We computed the closed-form solution analytically and we verified our results using automatic differentiation tools on large number of conditions including synthetic and real data. This has been another method to verify the correctness of the derivative formulas.

DIVA experimental details

DIVA Extend For Table 2 and Figure 4 we first split the original training set into 50% training subset and 50% pool subset that will be used to selectively extend the training subset with DIVA or other extension approaches. We run DIVA Extend LOO and uniform sampling to add pool samples to the training set. In both settings we incrementally extend the training subset from the pool (For DIVA, by selecting samples with top DIVA score) in each step we extend an equal number of samples ((# pool samples) // (# number of steps)). In the figure we present the test error as a function of training set size and compare DIVA sample selection with selecting the same number of samples at uniform from the pool set. In Table 2, we present the improvement in test accuracy at the 50% extend mark of the pool set (e.g. extending 25% of the original training set) between DIVA and uniform sample selection of the same number of samples. For both experiments we use the ResNet-34 architecture.

DIVA Reweigh In Table 1 we use DIVA Reweight LOO with the same parameters for all of the datasets we consider. The DIVA parameters we use are $K = 4$ for the number of steps and $\eta = 0.15$ for the step-size. As with DIVA Extend, we use ResNet 34 for the architecture for the representation.

DIVA validation loss We find the cross entropy loss to work better as the loss function applied to the validation predictions (both in LOO and regular validation). Further for LOO we find it crucial to apply the validation loss only on mis-classified LOO predictions to improve the test accuracy of the model, this can be interpreted as the cross entropy loss with a "hard margin hinge" loss.

Outlier rejection For the results presented in the side-by-side Table and Figure 3 we apply 20% random label noise to each class in the dataset and use DIVA LOO to compute the normalized DIVA gradient (DIVA score) of each sample. The F1 score reports classification by thresholding with $\epsilon = 0$ and the AUC is computing by thresholds spanning detection of no samples, to detection of all samples.

C Proofs of Propositions

We write the proposition statements for convenience.

Proposition (Model-Dataset Derivative $\nabla_\alpha \mathbf{w}_\alpha$)**.** *For the ridge regression problem eq. (5) and* $\mathbf{w}_\alpha$ *defined as in eq. (6), define*

$$\mathbf{C}_\alpha = (\mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Z} + \lambda \mathbf{I})^{-1}. \tag{16}$$

Then the Jacobian of $\mathbf{w}_\alpha$ *with respect to* α *is given by*

$$\nabla_\alpha \mathbf{w}_\alpha = \mathbf{Z}\mathbf{C}_\alpha \circ \big((\mathbf{I} - \mathbf{Z}\mathbf{C}_\alpha \mathbf{Z}^\top \mathbf{D}_\alpha)\mathbf{Y}\big). \tag{17}$$

Proof of Proposition 1:
We recall the settings of the problem are $\mathbf{Z} \in \mathbb{R}^{n \times m}$, $\mathbf{Y} \in \mathbb{R}^{n \times k}$, $\mathbf{w}_\alpha \in \mathbb{R}^{m \times k}$ and the importance weights $\alpha \in \mathbb{R}^n$. Here n is the number of samples, m is the number of parameters for each output of the model, and k is the number of classes (represented via the one-hot convention).

The derivation of $\nabla_\alpha \mathbf{w}_\alpha$ we present is computational in nature and without the loss of generality we consider the derivative for a single output class $k = 1$ (one-vs-all classification naturally extends). For the single class settings, the relevant dimensions are $\mathbf{w}_\alpha \in \mathbb{R}^m$ and $\nabla_\alpha \mathbf{w}_\alpha \in \mathbb{R}^{m \times n}$ (for numerator layout convention of the derivative). To further simplify we consider the derivative entrywise for single index α_r,

$$\frac{\partial \mathbf{w}_\alpha}{\partial \alpha_r} \in \mathbb{R}^{m \times 1}.$$

The closed-form solution is,

$$\mathbf{w}_\alpha = \mathbf{C}_\alpha \mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Y}.$$

using chain rule,

$$\frac{\partial \mathbf{w}_\alpha}{\partial \alpha_r} = \frac{\partial \mathbf{C}_\alpha}{\partial \alpha_r} \times \mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Y} + \mathbf{C}_\alpha \times \frac{\partial \mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Y}}{\partial \alpha_r}.$$

We compute the two parts of the derivative in turn:
Let $\mathbf{K}_\alpha = (\mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Z} + \lambda \mathbf{I})$ so that $\mathbf{K}_\alpha = \mathbf{C}_\alpha^{-1}$. Then

$$\frac{\partial \mathbf{K}_\alpha}{\partial \alpha_r} = \frac{\partial}{\partial \alpha_r}(\mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Z} + \lambda \mathbf{I}) = \mathbf{Z}_{r,:}^\top \mathbf{Z}_{r,:}$$

where $\mathbf{Z}_{r,:} \in \mathbb{R}^{1 \times m}$ is the rth row of $\mathbf{Z}$. Next note,

$$\mathbf{C}_\alpha \mathbf{K}_\alpha = \mathbf{I}$$

differentiating both sides,

$$\frac{\partial \mathbf{C}_\alpha \mathbf{K}_\alpha}{\partial \alpha_r} = \mathbf{0}.$$

Applying chain rule we get,

$$\frac{\partial \mathbf{C}_\alpha \mathbf{K}_\alpha}{\partial \alpha_r} = \frac{\partial \mathbf{C}_\alpha}{\partial \alpha_r} \mathbf{K}_\alpha + \mathbf{C}_\alpha \frac{\partial \mathbf{K}_\alpha}{\partial \alpha_r},$$

rearranging, substituting $\frac{\partial \mathbf{K}_\alpha}{\partial \alpha_r}$, and multiplying to the right by $\mathbf{C}_\alpha$

$$\frac{\partial \mathbf{C}_\alpha}{\partial \alpha_r} = -\mathbf{C}_\alpha \mathbf{Z}_{r,:}^\top \mathbf{Z}_{r,:} \mathbf{C}_\alpha.$$

Next, by direct computation $\frac{\partial \mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Y}}{\partial \alpha_r}$ satisfies

$$\frac{\partial \mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Y}}{\partial \alpha_r} = y_r \cdot \mathbf{Z}_{r,:}^\top.$$

Combining the original terms we have

$$\begin{aligned}\frac{\partial \mathbf{w}_\alpha}{\partial \alpha_r} &= -\big(\mathbf{C}_\alpha \mathbf{Z}_{r,:}^\top \mathbf{Z}_{r,:} \mathbf{C}_\alpha\big) \times \mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Y} + y_r \cdot \mathbf{C}_\alpha \times \mathbf{Z}_{r,:}^\top \\ &= \mathbf{C}_\alpha \mathbf{Z}_{r,:}^\top \big(y_r - \mathbf{Z}_{r,:} \mathbf{C}_\alpha \mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Y}\big).\end{aligned}$$

Now the full derivative is written as

$$\boxed{\nabla_\alpha \mathbf{w}_\alpha = \mathbf{Z}\mathbf{C}_\alpha \circ \big((\mathbf{I} - \mathbf{Z}\mathbf{C}_\alpha \mathbf{Z}^\top \mathbf{D}_\alpha)\mathbf{Y}\big)}$$

Proposition (Validation Loss Dataset Derivative)**.** *Define* $\mathbf{L}$ *as the matrix of the loss function derivative with respect to network training outputs as,*

$$\mathbf{L} = \left[\frac{\partial \ell}{\partial f}(f_{\mathbf{w}_\alpha}(\mathbf{x}_1), y_1), \cdots \frac{\partial \ell}{\partial f}(f(\mathbf{x}_N), y_N)\right].$$

Then the dataset derivative of the importance weights with respect to final validation loss is given by

$$\nabla_\alpha L_{val}(\mathbf{w}_\alpha) = \mathbf{Z}\mathbf{C}_\alpha \mathbf{Z}^\top \times \big(\mathbf{L}^\top \mathbf{Y}^\top (\mathbf{I} - \mathbf{D}_\alpha \mathbf{Z}\mathbf{C}_\alpha \mathbf{Z}^\top)\big). \tag{18}$$

Proof of Proposition 2:
This follows from the chain rule combined with simplification of broadcasting terms. We again consider the single output settings with the single coordinate derivative $\frac{\partial L_{\text{val}}}{\partial \alpha_r}$ which is given as,

$$\frac{\partial L_{\text{val}}}{\partial \alpha_r} = \nabla_{\mathbf{w}} L_{\text{val}} \frac{\partial \mathbf{w}_\alpha}{\partial \alpha_r}.$$

With

$$\begin{aligned}&= \sum_{i=1}^{n} \mathbf{L}_{:,i} \times \mathbf{z}_i^\top \\ &= \mathbf{L}^\top \mathbf{Z}.\end{aligned}$$

Therefore

$$\left(\nabla_\alpha L_{\text{val}}(\mathbf{w}_\alpha)\right)_i = \sum_{j,k} \left(\nabla_\alpha \mathbf{w}_\alpha\right)_{i,j,k} (\mathbf{L}^\top \mathbf{Z})_{j,k}.$$

Now $\mathbf{L}, \mathbf{Z}$ can be separated into the two terms of $\nabla_\alpha \mathbf{w}_\alpha$,

$$\boxed{\nabla_\alpha L_{\text{val}}(\mathbf{w}_\alpha) = \mathbf{Z}\mathbf{C}_\alpha\mathbf{Z}^\top \times \left(\mathbf{L}^\top\mathbf{Y}^\top(\mathbf{I} - \mathbf{D}_\alpha\mathbf{Z}\mathbf{C}_\alpha\mathbf{Z}^\top)\right)} \tag{19}$$

Next we consider the derivations for the Leave One Out (LOO) framework. In the LOO framework one applies cross-validation to a training set $\{(\mathbf{z}_1, y_1), \ldots (\mathbf{z}_n, y_n)\}$ by running n-fold cross validation, where in each fold, the ith sample $(\mathbf{z}_i, y_i)$ is taken out and is used for validation while the optimal classifier is solved for the remaining of the training task,

$$\mathbf{w}_{-i}, = \arg\min_{\mathbf{w}} \sum_{j \neq i} \ell(f(\mathbf{z}_j), y_j). \tag{20}$$

Then the LOO prediction at the ith index is defined as $(\mathbf{f}_{\text{LOO}})_i = f_{\mathbf{w}_{-i}}(\mathbf{z}_i)$. Below we prove the LOO predictions can be written in closed-form without explicit cross validation calculations.

Proposition 4 (Closed-form LOO prediction vector)**.** *Define the LOO vector predictions as,*

$$\mathbf{f}_{\text{LOO}} = [f_{\mathbf{w}_{-1}}(\mathbf{z}_1), \ldots, f_{\mathbf{w}^{-n}}(\mathbf{z}_n)]^\top$$

and define

$$\mathbf{R} = \mathbf{Z}^\top(\mathbf{Z}^\top\mathbf{Z} + \lambda\mathbf{I})^{-1}\mathbf{Z}$$

then for the learning task eq. (20) LOO predictions are given as

$$\mathbf{f}_{\text{LOO}} = \frac{\mathbf{R}\mathbf{y} - \mathit{diag}(\mathbf{R})\mathbf{y}}{\mathbf{I} - \mathit{diag}(\mathbf{R})}. \tag{21}$$

Proof of Proposition 4:
The proof is reproduced from Rifkin & Lippert (2007) for completeness.
Without the loss of generality we derive the LOO prediction of $\mathbf{z}_n$. Namely given, $\{(\mathbf{z}_1, y_1), \ldots (\mathbf{z}_n, y_n)\}$ we use $\{(\mathbf{z}_1, y_1), \ldots (\mathbf{z}_{n_1}, y_{n-1})\}$ for training and validate using $\{(\mathbf{z}_n, y_n\}$. Denote $\mathbf{w}^{-n}$ as be the optimal solution to this training task with regularization parameter λ and define the (currently unknown) LOO prediction as

$$q = f_{\mathbf{w}^{-n}}(\mathbf{z}_n).$$

We define the modified learning task consisting of $\{(\mathbf{z}_1, y_1), \ldots (\mathbf{z}_{n-1}, y_{n-1}), (\mathbf{z}_n, q)\}$ where we added the data point $(\mathbf{z}_n, q)$. Note that the optimal solution with λ regularization to the modified learning task is again $\mathbf{w}^{-n}$ since q is taken to have a zero residual. therefore the solution to the modified learning problem can be written in closed-form as,

$$\mathbf{w}^{-n} = (\mathbf{Z}^\top\mathbf{Z} + \lambda\mathbf{I})^{-1}\mathbf{Z}^\top[\mathbf{y}_{1:n-1}, q]^\top.$$

Using $\mathbf{w}^{-n}$ we write the equation for the LOOV prediction q,

$$q = \langle \mathbf{z}_n, \mathbf{w}^{-i} \rangle = \mathbf{z}_n^\top(\mathbf{Z}^\top\mathbf{Z} + \lambda\mathbf{I})^{-1}\mathbf{Z}^\top[\mathbf{y}_{1:n-1}, q]^\top.$$

Let $\mathbf{R} = \mathbf{Z}(\mathbf{Z}^\top\mathbf{Z} + \lambda\mathbf{I})^{-1}\mathbf{Z}^\top$ then we have

$$q = \mathbf{z}_n^\top\mathbf{w}_{-i} = \mathbf{R}_{n,:}[\mathbf{Y}_{1:n-1}, q]^\top$$

and by adding and subtracting $\mathbf{R}_{:,n}$ multiplied by $[0, y_n]^\top$ we get,

$$q = \mathbf{z}_n^\top\mathbf{w}_{-i} = \mathbf{R}_{:n-1,n}[\mathbf{Y}_{1:n-1}]^\top + \mathbf{R}_{:,n}[0, q] + \mathbf{R}_{:,n}[0, y_n]^\top - \mathbf{R}_{:,n}[0, y_n]^\top.$$

Re-arranging we have

$$q - \mathbf{R}_{nn}q = \mathbf{R}_{:,n}\mathbf{y} - y_n\mathbf{R}_{n,n}$$

Solving for q we get,

$$q = \frac{\mathbf{R}_{:,n}\mathbf{y} - y_n\mathbf{R}_{n,n}}{1 - \mathbf{R}_{n,n}}$$

And without the loss of generality the full prediction vector is given as,

$$\boxed{\mathbf{f}_{\text{LOO}} = \frac{\mathbf{R}\mathbf{y} - \text{diag}(\mathbf{R})\mathbf{y}}{\text{diag}(\mathbf{I} - \mathbf{R})}} \tag{22}$$

Given the closed-form LOOV expression we may use $\mathbf{f}_{\text{LOO}}$ for the validation loss to compute the dataset derivative on L_{val} without any additional validation data. While this may seem contradictory as we are optimizing the dataset validated via the weighting duality between sample loss weighting and data scaling, we define the leave one out value predictions in the weighted dataset settings and evaluate on the original (unweighted) data points as,

$$f_{\mathbf{w}_\alpha^{-i}}(\mathbf{z}_i) \tag{23}$$

with,

$$\mathbf{w}_\alpha^{-i} = \arg\min_{\mathbf{w}} \sum_{j \neq i} \alpha_j \ell(f(\mathbf{z}_j), y_j). \tag{24}$$

Therefore the α-weighted LOO term is $f_{\mathbf{w}_\alpha^{-i}}(\mathbf{z}_i)$ is faithful to the original distribution despite being trained with the weighted loss eq. (24). We in fact are able to show that α-weighted LOO formulation also admits a closed-form solution that satisfies our definition and for DIVA LOO we utilize the derivative of the closed-form to optimize the dataset.

Proposition. *Define*

$$\mathbf{R}_\alpha = \mathbf{Z}^\top \sqrt{\mathbf{D}_\alpha}(\mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Z} + \lambda \mathbf{I})^{-1} \sqrt{\mathbf{D}_\alpha}\mathbf{Z}$$

Then the α-weighted LOOV predictions defined in eq. (12) admit a closed-form solution:

$$f_{\mathbf{w}_\alpha^{-i}}(\mathbf{z}_i) = \left[\frac{\mathbf{R}_\alpha \sqrt{\mathbf{D}_\alpha}\mathbf{Y} - \textit{diag}(\mathbf{R}_\alpha)\sqrt{\mathbf{D}_\alpha}\mathbf{Y}}{\textit{diag}(\sqrt{\mathbf{D}_\alpha} - \sqrt{\mathbf{D}_\alpha}\mathbf{R}_\alpha)} \right]_i, \tag{25}$$

Further the LOO dataset derivative is well-defined and satisfies the following gradient condition,

$$\textit{diag}(\nabla_\alpha \mathbf{f}_{\textit{LOO}}) = \mathbf{0}. \tag{26}$$

The gradient condition $\text{diag}(\nabla_\alpha \mathbf{f}_{\text{LOO}}) = \mathbf{0}$ implies that the LOO prediction at $\mathbf{z}_i$ does not depend on α_i and ensures that the closed-form solution is well defined in the α-weighted settings and is differentiable.

Proof of Proposition 3:

The proof builds on Proposition 4 for the weighted settings.

In general since LOO expression describes the weighting problem, it must be shown that the introduction of the weights do not break the argument of Proposition 4. Considering the optimization problem in eq. (12). We also use the duality between the loss weights and data scaling to note that $\mathbf{w}_\alpha^- i$ can be derived by considering the unweighted LOO eq. (20) with the modified dataset,

$$\{(\sqrt{\alpha_1}\mathbf{z}_1, \sqrt{\alpha_1}y_1), \ldots (\sqrt{\alpha_n}\mathbf{z}_n, \sqrt{\alpha_n}y_n)\}. \tag{27}$$

Indeed in this settings with the newly defined data the derivation in Proposition 4 of the final prediction vector of the LOO entries holds with the same optimal solution $\mathbf{w}_\alpha$ due to the duality between data scaling and loss weights. Nonetheless the closed-form LOO predictions $\mathbf{f}_{\text{LOO}}$ are evaluated at the data-points $\sqrt{\alpha_i}\mathbf{z}_i$. Since the model is linear the final predictions at the original data-points of the weighted training settings are written as $\mathbf{f}_{\text{LOO}}/\sqrt{\alpha}$.

Noting the weighted training problem can be expressed as $\tilde{\mathbf{Y}} = \sqrt{\mathbf{D}_\alpha}\mathbf{Y}$, $\tilde{\mathbf{Z}} = \sqrt{\mathbf{D}_\alpha}\mathbf{Z}$ we use Proposition 4 to write analogously

$$\mathbf{R}_\alpha = \mathbf{Z}^\top \sqrt{\mathbf{D}_\alpha}(\mathbf{Z}^\top \mathbf{D}_\alpha \mathbf{Z} + \lambda \mathbf{I})^{-1} \sqrt{\mathbf{D}_\alpha}\mathbf{Z} \tag{28}$$

and divide by $\sqrt{\alpha}$ by multiplying the denominator by $\sqrt{\mathbf{D}_\alpha}$,

$$f_{\mathbf{w}_\alpha^{-i}}(\mathbf{z}_i) = \left[\frac{\mathbf{R}_\alpha \sqrt{\mathbf{D}_\alpha}\mathbf{Y} - \text{diag}(\mathbf{R}_\alpha)\sqrt{\mathbf{D}_\alpha}\mathbf{Y}}{\text{diag}(\sqrt{\mathbf{D}_\alpha} - \sqrt{\mathbf{D}_\alpha}\mathbf{R}_\alpha)} \right]_i, \tag{29}$$

Since the derivation at the ith index of the weighted LOO prediction, $(\mathbf{f}_{\text{LOO}})_i = (\mathbf{w}_\alpha^{-i})^\top \mathbf{z}_i$ is entirely independent of α_i, we have

$$\frac{\partial (f_{\mathbf{w}_\alpha^{-i}}(\mathbf{z}_i))_i}{\partial \alpha_i} = 0.$$