

# IF INFLUENCE FUNCTIONS ARE THE ANSWER, THEN WHAT IS THE QUES- TION?

Paper review

---

Alex Ali Maleknia

# CONTENTS

---

1. Setting
2. Problems
3. Response functions
4. Error decomposition
5. Summary

Consider a prediction task (regression problem) with:

- Input space  $\mathcal{X}$ ;
- Output space  $\mathcal{Y}$ ;
- Training set  $\mathcal{D}^n = \{z_i\}_{i=1}^n$  where  $z_i = (x_i, y_i)$  for all  $i = 1, \dots, n$ ,  
 $X = (x_1, \dots, x_n)$ ,  $Y = (y_1, \dots, y_n)$ ;
- Parameter  $\theta \in \Theta := \mathbb{R}^d$ ;
- $f(\theta; x)$  estimator of  $\mathcal{Y}|X$ ;
- $l : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$  loss function (e.g.,  $l(y', y) \mapsto \|y' - y\|^2$ ).

We aim to minimize the training error:

$$L(\theta; \mathcal{D}^n) = \frac{1}{n} \sum_{i=1}^n l(f(\theta; x_i), y_i).$$

What happens if we change the importance of a training point  $z = (x, y)$  of the dataset?

Call  $\hat{\theta} = \arg \min_{\theta \in \Theta} L(\theta; \mathcal{D}^n)$ .

How different is it from

$$\hat{\theta}_{\varepsilon, -z} = \arg \min_{\theta \in \Theta} (L(\theta; \mathcal{D}^n) - \varepsilon l(f(\theta; x), y)) \quad ?$$

We can re-train the whole model on  $\mathcal{D}^n \setminus \{z\}$  (Leave One Out method), or...

## Definition

Given  $(\bar{x}, \bar{y}) = \bar{z} \in \mathcal{D}^n$ , the *influence loss difference* relative to  $\bar{z}$  is:

$$\mathcal{Q}(\bar{z}) = \left. \frac{d}{d\varepsilon} [L(\theta; \mathcal{D}^n) - \varepsilon l(f(\theta; \bar{x}), \bar{y})] \right|_{\varepsilon = \frac{1}{n}}$$

**Interpretation:** It indicates how much the training error changes when we remove a training data  $\bar{z}$ .

## Definition

Given  $(\bar{x}, \bar{y}) = \bar{z} \in \mathcal{D}^n$ , the *influence function* relative to  $\bar{z}$  is:

$$\mathcal{J}(\bar{z}) = \lim_{\varepsilon \downarrow \frac{1}{n}} \frac{\hat{\theta}_{\varepsilon, -\bar{z}} - \hat{\theta}_{1/n, -\bar{z}}}{\varepsilon}$$

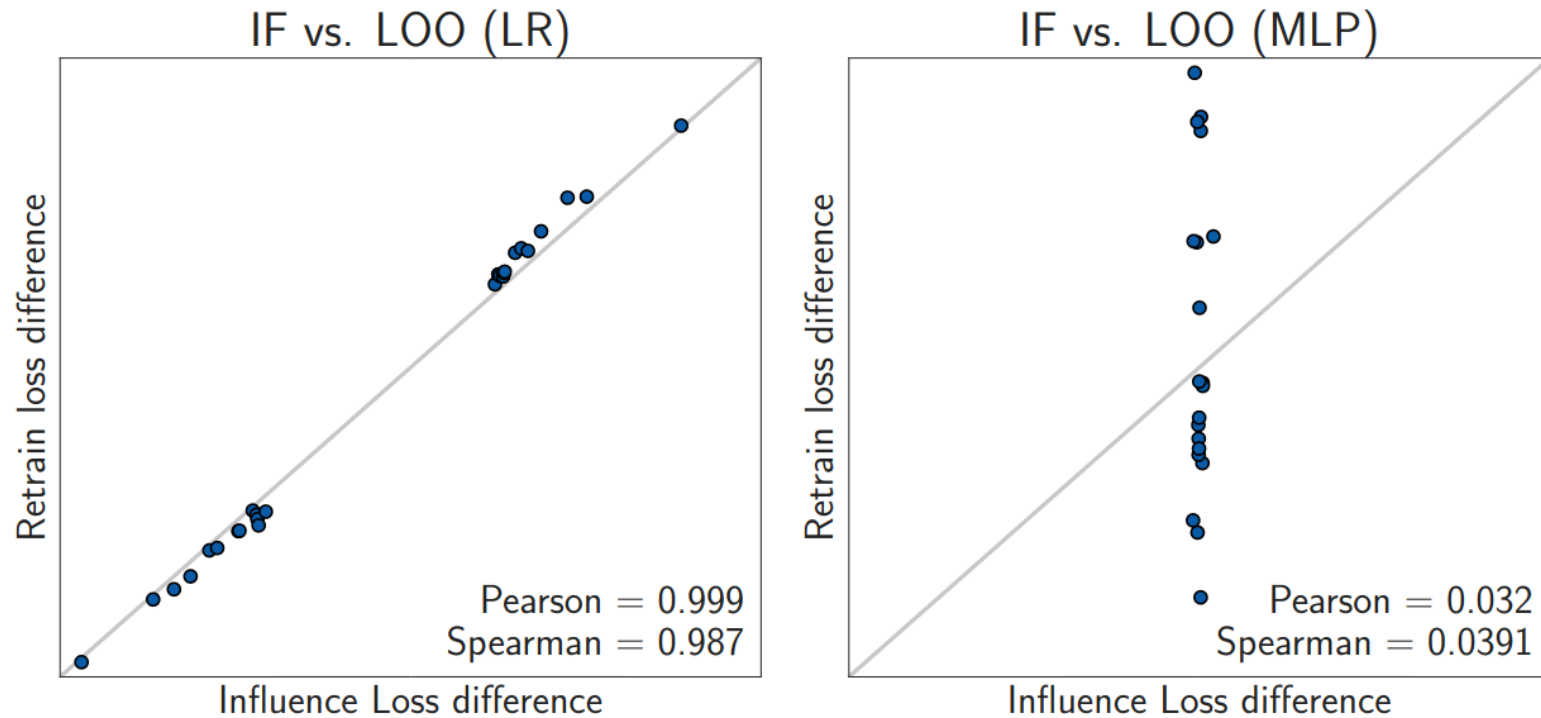
**Interpretation:** It represents the direction in which the optimal parameter moves when the training error is changed by removing the training data  $\bar{z}$ .

Assume  $L$  is strongly convex. Evaluating influence functions requires heavy computations:

$$\mathcal{J}(z) = \frac{1}{n} H_{\hat{\theta}}^{-1} \nabla l(f(\hat{\theta}; x), y),$$

$$\mathcal{Q}(z) = \frac{1}{n} \nabla l(f(\hat{\theta}; x), y)^\top H_{\hat{\theta}}^{-1} \nabla l(f(\hat{\theta}; x), y),$$

where  $H_{\hat{\theta}}$  is the Hessian of  $L$ , which can be difficult to compute.



The strong convexity assumption is essential!



**Solution 1.** For iHVP, there are good approximations that only require  $O(nd)$  flops instead of  $O(n^3)$ .

**Solution 2.** Change point of view: Influence functions are not approximators of LOO retraining, but instead of the proximal Bregman response function (PBRF).

We define the response function in the general setting as:

$$\hat{r}_z(\varepsilon) = \arg \min_{\theta \in \Theta} (L(\theta; \mathcal{D}^n) - \varepsilon l(f(\theta; x), y)).$$

Note that  $\hat{r}_z(\varepsilon) = \hat{\theta}_{\varepsilon, -z}$  and  $\hat{r}_z(0) = \hat{\theta}$ . Since (by IFT)  $\hat{r}$  is differentiable at 0, we can define the influence functions as first order approximant of  $\hat{r}$ . In fact, expanding with Taylor near 0 we get:

$$\hat{r}_{z, \text{lin}}\left(\frac{1}{n}\right) = \hat{r}_z(0) + \left. \frac{d\hat{r}_z}{d\varepsilon} \right|_{\varepsilon=0} (\varepsilon - 0) = \hat{\theta} + \frac{1}{n} H_{\hat{\theta}}^{-1} \nabla l(f(\hat{\theta}; x), y).$$

We need  $H_{\theta}$  to be positive definite in order to invert it, so  $\theta$  must be a minimum point.

In order for the influence functions to be computable in the MLP case, we need to address the hessian inverse. This can be done by approximating  $H_{\hat{\theta}}$  with the Gauss-Newton Hessian (GNH) and adding a damping term to ensure GNH is invertible:

$$\mathcal{J}^\dagger(z) = \frac{1}{n} \left( J_{y\hat{\theta}}^\top H_{\hat{\theta}} J_{y\hat{\theta}} + \lambda \mathbf{I} \right)^{-1} \nabla l(f(\hat{\theta}; x), y),$$

where  $J_{y\hat{\theta}}$  is the Jacobian of  $F(\theta) = (f(\theta; x_1), \dots, f(\theta; x_n))$  in  $\hat{\theta}$ .

We can get the previous formula by linearizing near 0:

$$\hat{r}_{z,\text{damp}}(\varepsilon) = \arg \min_{\theta \in \Theta} L(\theta; \mathcal{D}^n) - \varepsilon l(f(\theta; x), y) + \frac{\lambda}{2} \|\theta - \hat{\theta}\|^2,$$

$$\hat{r}_{z,\text{damp},\text{lin}}(1/n) \approx \hat{\theta} + \mathcal{J}^\dagger.$$

In practice,  $\theta$  is not a minimum point for  $L$ .

However, we can consider another training error for which the early arrested parameter  $\theta^s$  is optimal:

$$\mathcal{L}(\theta; \theta^s, \mathcal{D}^n) = \frac{1}{n} \sum_{i=1}^n D_{l^{(i)}}(f(\theta; x_i), f(\theta^s; x_i)),$$

where  $D_{l^{(i)}}(y, y') = l(y, y_i) - l(y', y_i) - \nabla_1 l(y', y_i)^\top (y - y')$  is called Bregman difference.

We can then define the PBRF as:

$$r_{z, \text{damp}}^b(\varepsilon) = \arg \min_{\theta \in \Theta} \mathcal{L}(\theta; \theta^s, \mathcal{D}^n) - \varepsilon l(f(\theta, x), y) + \frac{\lambda}{2} \|\theta - \theta^s\|^2.$$

The optimal solution for the linearised PBRF is the same as the influence function estimation:

$$r_{z, \text{damp}, \text{lin}}^b(1/n) = \theta^s + \mathcal{J}^\dagger(z).$$

Therefore, influence functions do NOT depict the retraining with LOO algorithm using  $L$  as training error.

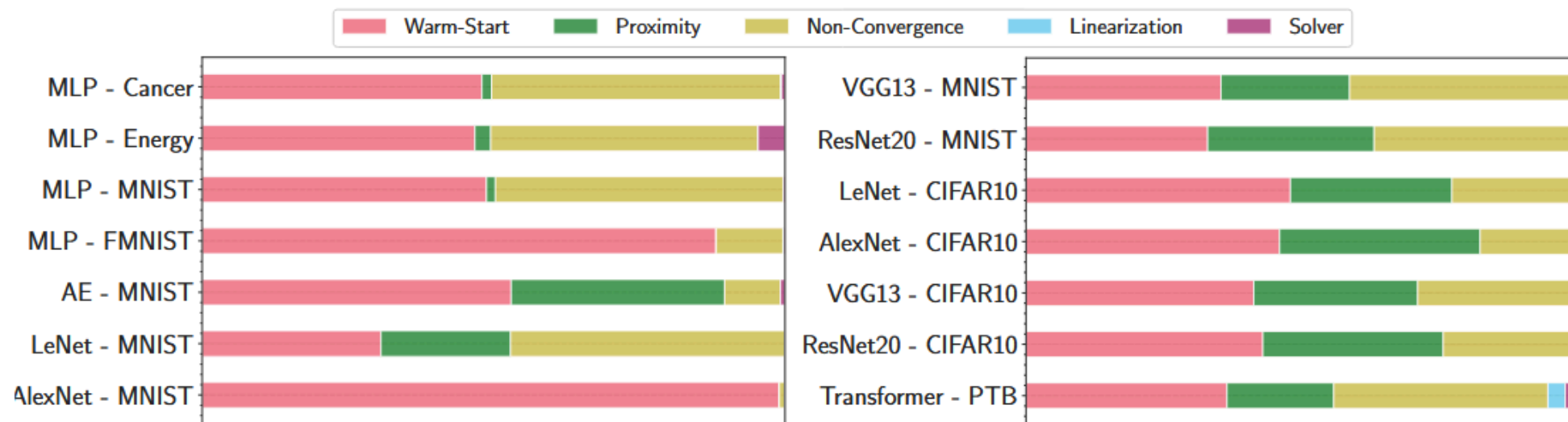
Instead they estimate what happens after training from  $\theta^s$  using as empirical risk:

$$\mathcal{L}(\theta; \theta^s, \mathcal{D}^n) - \frac{1}{n}l(f(\theta, x), y) + \frac{\lambda}{2}\|\theta - \theta^s\|^2.$$

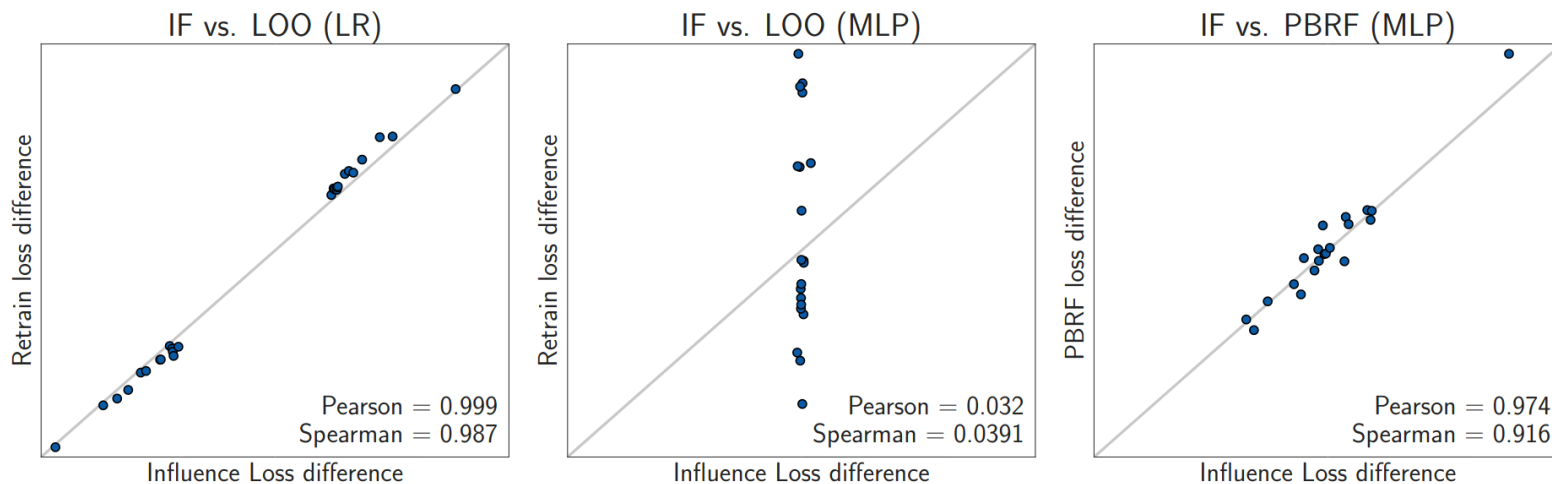
We can decompose the approximation error of the influence functions in 5 categories:

- **Warm-start gap:** LOO starts from a random parameter (cold start), while IF are related to  $\theta^s$ ; we can then converge to another “optimal” point;
- **Proximity gap:** the factor  $\|\theta - \theta^s\|$  induces the warm start not to move far away from  $\theta^s$ ;
- **Non-convergence gap:** in practice we almost never start from a fully trained network;
- **Linearization error:** produced by approximating the Taylor expansion at first order;
- **Solver error:** algorithms used to compute iHVP are approximated.

The PBRF method annihilates the first three components.



Influence functions seem not to work well for NNs, as the loss function is non-linear. In reality, they are approximating the result of PBRF instead of LOO retraining.



[“If Influence Functions are the Answer, Then What is the Question?”, J. Bae, N. Ng, A. Lo, M. Ghassemi, R. Grosse, 2022]

[“On the Accuracy of Influence Functions for Measuring Group Effects”, Koh et al., 2019]

[“Understanding Black-box Predictions via Influence Functions”, Koh and Liang, 2020]



Ciaos