

417 A Derivation of the influence function

418 In this section, we derive the influence function for linear models [20], as shown in Equation (3), and
 419 for neural networks [3], as shown in Equation (7). In particular, we reformulate the derivation by Bae
 420 et al. [3] to fit our setting.

421 A.1 Influence function for linear models

422 We begin by considering the re-weighted objective introduced in Equation (2):

$$\mathcal{J}(\theta, \epsilon) = \frac{1}{N} \sum_{(x, y) \in \mathcal{D}_{\text{train}}} \mathcal{L}(x, y, \theta) + \epsilon \mathcal{L}(x', y', \theta). \quad (15)$$

423 Assuming that the loss function \mathcal{L} is twice continuously differentiable and that $\mathcal{J}(\theta, \epsilon)$ is strictly
 424 convex in θ , the optimal parameter $\theta_{x', y', \epsilon}^*$ minimizing the objective satisfies the first-order optimality
 425 condition:

$$\nabla_{\theta} \mathcal{J}(\theta_{x', y', \epsilon}^*, \epsilon) = 0. \quad (16)$$

426 To justify that the response function $\theta_{x', y', \epsilon}^*$ is differentiable with respect to ϵ , we apply the Implicit
 427 Function Theorem to the optimality condition. The following conditions must be satisfied for the
 428 theorem to apply:

- 429 • The function $\nabla_{\theta} \mathcal{J}(\theta, \epsilon)$ is continuously differentiable in both θ and ϵ . This holds because
 430 $\mathcal{J}(\theta, \epsilon)$ is constructed as a linear combination of smooth loss functions, and its dependence
 431 on ϵ is linear.
- 432 • The optimality condition $\nabla_{\theta} \mathcal{J}(\theta_{x', y', \epsilon}^*, \epsilon) = 0$ holds by definition, since $\theta_{x', y', \epsilon}^*$ minimizes
 433 the objective $\mathcal{J}(\theta, \epsilon)$.
- 434 • The Hessian $\nabla_{\theta}^2 \mathcal{J}(\theta, \epsilon)$, taken with respect to θ , is non-singular in a neighborhood of $\epsilon = 0$,
 435 as $\mathcal{J}(\theta, \epsilon)$ is assumed to be strictly convex in θ .

436 Under these conditions, the Implicit Function Theorem ensures that $\theta_{x', y', \epsilon}^*$ is continuously differen-
 437 tiable with respect to ϵ , and we differentiate Equation (16) using the chain rule:

$$\frac{d}{d\epsilon} (\nabla_{\theta} \mathcal{J}(\theta_{x', y', \epsilon}^*, \epsilon)) = \nabla_{\theta}^2 \mathcal{J}(\theta_{x', y', \epsilon}^*, \epsilon) \cdot \frac{d\theta_{x', y', \epsilon}^*}{d\epsilon} + \nabla_{\theta} \mathcal{L}(x', y', \theta_{x', y', \epsilon}^*) = 0. \quad (17)$$

438 Solving for the derivative yields:

$$\frac{d\theta_{x', y', \epsilon}^*}{d\epsilon} = - (\nabla_{\theta}^2 \mathcal{J}(\theta_{x', y', \epsilon}^*, \epsilon))^{-1} \nabla_{\theta} \mathcal{L}(x', y', \theta_{x', y', \epsilon}^*). \quad (18)$$

439 Evaluating at $\epsilon = 0$, we obtain the influence function:

$$\left. \frac{d\theta_{x', y', \epsilon}^*}{d\epsilon} \right|_{\epsilon=0} = - (\nabla_{\theta}^2 \mathcal{J}(\theta^*, 0))^{-1} \nabla_{\theta} \mathcal{L}(x', y', \theta^*), \quad (19)$$

440 where $\theta^* := \theta_{x', y', \epsilon=0}^*$ is the minimizer of the original objective without perturbation.

441 A.2 Influence function for neural networks

442 Bae et al. [3] demonstrated that the influence function using the generalized Gauss-Newton Hessian
443 corresponds to that of the linearized form of the proximal Bregman response function objective³:

$$\theta_{\text{lin},x',y',\epsilon}^* := \arg \min_{\theta} \frac{1}{N} \sum_{(x,y) \in \mathcal{D}_{\text{train}}} D_{\mathcal{L}_{\text{quad}}} (h_x^{\text{lin},\theta}, h_x^{\theta_s}) + \frac{\lambda}{2} \|\theta - \theta_s\|^2 + \epsilon \nabla_{\theta} \mathcal{L}(x', \theta_s)^{\top} \theta, \quad (20)$$

444 where $h_x^{\theta} = g_{\theta}(x)$, and $\mathcal{L}_{\text{quad}}$ and $h_x^{\text{lin},\theta}$ are the quadratic and linear approximations of the loss and
445 model output, respectively:

$$\begin{aligned} \mathcal{L}_{\text{quad}}(h_x^{\theta}) &= \mathcal{L}(h_x^{\theta_s}) + \nabla_{h_x^{\theta}} \mathcal{L}(h_x^{\theta_s})^{\top} (h_x^{\theta} - h_x^{\theta_s}) + \frac{1}{2} (h_x^{\theta} - h_x^{\theta_s})^{\top} \nabla_{h_x^{\theta}}^2 \mathcal{L}(h_x^{\theta_s}) (h_x^{\theta} - h_x^{\theta_s}), \\ h_x^{\text{lin},\theta} &= h_x^{\theta_s} + \mathbf{J}_{h_x^{\theta} \theta_s} (\theta - \theta_s), \end{aligned} \quad (21)$$

446 where $\mathbf{J}_{h_x^{\theta} \theta_s} := \frac{\partial h_x^{\theta}}{\partial \theta} \big|_{\theta=\theta_s}$ is the Jacobian of the model output with respect to the parameters.

447 We now expand the Bregman divergence term. First, using $\mathcal{L}_{\text{quad}}(h_x^{\theta_s}) = \mathcal{L}(h_x^{\theta_s})$:

$$\begin{aligned} D_{\mathcal{L}_{\text{quad}}}(h_x^{\text{lin},\theta}, h_x^{\theta_s}) &= \mathcal{L}_{\text{quad}}(h_x^{\text{lin},\theta}) - \mathcal{L}_{\text{quad}}(h_x^{\theta_s}) - \nabla_{h_x^{\theta}} \mathcal{L}_{\text{quad}}(h_x^{\theta_s})^{\top} (h_x^{\text{lin},\theta} - h_x^{\theta_s}) \\ &= \mathcal{L}(h_x^{\theta_s}) + \nabla_{h_x^{\theta}} \mathcal{L}(h_x^{\theta_s})^{\top} (h_x^{\theta} - h_x^{\theta_s}) + \frac{1}{2} (h_x^{\theta} - h_x^{\theta_s})^{\top} \nabla_{h_x^{\theta}}^2 \mathcal{L}(h_x^{\theta_s}) (h_x^{\theta} - h_x^{\theta_s}) \\ &\quad - \mathcal{L}(h_x^{\theta_s}) - \nabla_{h_x^{\theta}} \mathcal{L}(h_x^{\theta_s})^{\top} (h_x^{\theta} - h_x^{\theta_s}). \end{aligned} \quad (22)$$

448 Next, using $h_x^{\text{lin},\theta} - h_x^{\theta_s} = \mathbf{J}_{h_x^{\theta} \theta_s} (\theta - \theta_s)$:

$$\begin{aligned} D_{\mathcal{L}_{\text{quad}}}(h_x^{\text{lin},\theta}, h_x^{\theta_s}) &= \nabla_{h_x^{\theta}} \mathcal{L}(h_x^{\theta_s})^{\top} \mathbf{J}_{h_x^{\theta} \theta_s} (\theta - \theta_s) + \frac{1}{2} (\theta - \theta_s)^{\top} \mathbf{J}_{h_x^{\theta} \theta_s}^{\top} \nabla_{h_x^{\theta}}^2 \mathcal{L}(h_x^{\theta_s}) \mathbf{J}_{h_x^{\theta} \theta_s} (\theta - \theta_s) \\ &\quad - \nabla_{h_x^{\theta}} \mathcal{L}(h_x^{\theta_s})^{\top} \mathbf{J}_{h_x^{\theta} \theta_s} (\theta - \theta_s) \\ &= \frac{1}{2} (\theta - \theta_s)^{\top} \mathbf{J}_{h_x^{\theta} \theta_s}^{\top} \nabla_{h_x^{\theta}}^2 \mathcal{L}(h_x^{\theta_s}) \mathbf{J}_{h_x^{\theta} \theta_s} (\theta - \theta_s). \end{aligned} \quad (23)$$

449 Since $\theta_{\text{lin},x',y',\epsilon}^*$ is the optimal solution, the gradient of the objective with respect to θ is zero at this
450 point:

$$0 = -\frac{1}{N} \sum_{(x,y) \in \mathcal{D}_{\text{train}}} \mathbf{J}_{h_x^{\theta} \theta_s}^{\top} \nabla_{h_x^{\theta}}^2 \mathcal{L}(h_x^{\theta_s}) \mathbf{J}_{h_x^{\theta} \theta_s} (\theta_{\text{lin},x',y',\epsilon}^* - \theta_s) + \lambda (\theta_{\text{lin},x',y',\epsilon}^* - \theta_s) + \epsilon \nabla_{\theta} \mathcal{L}(x', \theta_s). \quad (24)$$

451 Solving for $\theta_{\text{lin},x',y',\epsilon}^*$, we obtain:

$$\theta_{\text{lin},x',y',\epsilon}^* = \theta_s + (\mathbf{J}_{h_x^{\theta} \theta_s}^{\top} \mathbf{H}_{h_s} \mathbf{J}_{h_x^{\theta} \theta_s} + \lambda \mathbf{I})^{-1} \nabla_{\theta} \mathcal{L}(x', \theta_s) \epsilon, \quad (25)$$

452 where $\mathbf{J}_{h_x^{\theta} \theta_s}^{\top} \mathbf{H}_{h_s} \mathbf{J}_{h_x^{\theta} \theta_s} := \frac{1}{N} \sum_{(x,y) \in \mathcal{D}_{\text{train}}} \mathbf{J}_{h_x^{\theta} \theta_s}^{\top} \nabla_{h_x^{\theta}}^2 \mathcal{L}(h_x^{\theta_s}) \mathbf{J}_{h_x^{\theta} \theta_s}$ is the generalized Gauss-Newton
453 Hessian.

³For notational simplicity, we omit the label y in expressions involving the loss function or the Bregman divergence, as it is clear from context.

454 B Gradient derivation for edge-edit PBRF

455 Similar to the derivation of the influence function for the PBRF objective in Appendix A.2, we
 456 demonstrate that the influence function in Equation (10) corresponds to that of the linearized form of
 457 the edge-edit PBRF objective:

$$\begin{aligned} \theta_{\text{lin},\epsilon}^* := \arg \min_{\theta} \frac{1}{N} \sum_{v \in \mathcal{V}_{\text{train}}} D_{\mathcal{L}_{\text{quad}}} (h_v^{\mathcal{G},\theta,\text{lin}}, h_v^{\mathcal{G},\theta_s}) + \frac{\lambda}{2} \|\theta - \theta_s\|^2 \\ + \sum_{v \in \mathcal{V}_{\text{train}}} \epsilon \left(\nabla_{\theta} \mathcal{L}(h_v^{\mathcal{G},\theta_s}) - \nabla_{\theta} \mathcal{L}(h_v^{\mathcal{G}^{-\frac{1}{N}},\theta_s}) \right)^{\top} \theta, \end{aligned} \quad (26)$$

458 where $\mathcal{L}_{\text{quad}}$ and $h_v^{\mathcal{G},\theta,\text{lin}}$ denote the quadratic and linear approximations of the loss and the model
 459 output, respectively:

$$\begin{aligned} \mathcal{L}_{\text{quad}}(h_v^{\mathcal{G},\theta}) &= \mathcal{L}(h_v^{\mathcal{G},\theta_s}) + \nabla_{h_v^{\mathcal{G},\theta}} \mathcal{L}(h_v^{\mathcal{G},\theta_s})^{\top} (h_v^{\mathcal{G},\theta} - h_v^{\mathcal{G},\theta_s}) \\ &\quad + \frac{1}{2} (h_v^{\mathcal{G},\theta} - h_v^{\mathcal{G},\theta_s})^{\top} \nabla_{h_v^{\mathcal{G},\theta}}^2 \mathcal{L}(h_v^{\mathcal{G},\theta_s}) (h_v^{\mathcal{G},\theta} - h_v^{\mathcal{G},\theta_s}), \\ h_v^{\mathcal{G},\theta,\text{lin}} &= h_v^{\mathcal{G},\theta_s} + \mathbf{J}_{h_v^{\mathcal{G},\theta}} (\theta - \theta_s), \end{aligned} \quad (27)$$

460 where $\mathbf{J}_{h_v^{\mathcal{G},\theta}} := \frac{\partial h_v^{\mathcal{G},\theta}}{\partial \theta} \big|_{\theta=\theta_s}$ is the Jacobian of the model output with respect to the parameters.

461 We first expand the Bregman divergence term:

$$\begin{aligned} D_{\mathcal{L}_{\text{quad}}}(h_v^{\mathcal{G},\theta,\text{lin}}, h_v^{\mathcal{G},\theta_s}) &= \mathcal{L}_{\text{quad}}(h_v^{\mathcal{G},\theta,\text{lin}}) - \mathcal{L}_{\text{quad}}(h_v^{\mathcal{G},\theta_s}) - \nabla_{h_v^{\mathcal{G},\theta}} \mathcal{L}_{\text{quad}}(h_v^{\mathcal{G},\theta_s})^{\top} (h_v^{\mathcal{G},\theta,\text{lin}} - h_v^{\mathcal{G},\theta_s}) \\ &= \mathcal{L}(h_v^{\mathcal{G},\theta_s}) + \nabla_{h_v^{\mathcal{G},\theta}} \mathcal{L}(h_v^{\mathcal{G},\theta_s})^{\top} (h_v^{\mathcal{G},\theta} - h_v^{\mathcal{G},\theta_s}) \\ &\quad + \frac{1}{2} (h_v^{\mathcal{G},\theta} - h_v^{\mathcal{G},\theta_s})^{\top} \nabla_{h_v^{\mathcal{G},\theta}}^2 \mathcal{L}(h_v^{\mathcal{G},\theta_s}) (h_v^{\mathcal{G},\theta} - h_v^{\mathcal{G},\theta_s}) \\ &\quad - \mathcal{L}(h_v^{\mathcal{G},\theta_s}) - \nabla_{h_v^{\mathcal{G},\theta}} \mathcal{L}(h_v^{\mathcal{G},\theta_s})^{\top} (h_v^{\mathcal{G},\theta} - h_v^{\mathcal{G},\theta_s}). \end{aligned} \quad (28)$$

462 Using $h_v^{\mathcal{G},\theta,\text{lin}} - h_v^{\mathcal{G},\theta_s} = \mathbf{J}_{h_v^{\mathcal{G},\theta}} (\theta - \theta_s)$, we substitute into the expression:

$$\begin{aligned} D_{\mathcal{L}_{\text{quad}}}(h_v^{\mathcal{G},\theta,\text{lin}}, h_v^{\mathcal{G},\theta_s}) &= \nabla_{h_v^{\mathcal{G},\theta}} \mathcal{L}(h_v^{\mathcal{G},\theta_s})^{\top} \mathbf{J}_{h_v^{\mathcal{G},\theta}} (\theta - \theta_s) \\ &\quad + \frac{1}{2} (\theta - \theta_s)^{\top} \mathbf{J}_{h_v^{\mathcal{G},\theta}}^{\top} \nabla_{h_v^{\mathcal{G},\theta}}^2 \mathcal{L}(h_v^{\mathcal{G},\theta_s}) \mathbf{J}_{h_v^{\mathcal{G},\theta}} (\theta - \theta_s) \\ &\quad - \nabla_{h_v^{\mathcal{G},\theta}} \mathcal{L}(h_v^{\mathcal{G},\theta_s})^{\top} \mathbf{J}_{h_v^{\mathcal{G},\theta}} (\theta - \theta_s) \\ &= \frac{1}{2} (\theta - \theta_s)^{\top} \mathbf{J}_{h_v^{\mathcal{G},\theta}}^{\top} \nabla_{h_v^{\mathcal{G},\theta}}^2 \mathcal{L}(h_v^{\mathcal{G},\theta_s}) \mathbf{J}_{h_v^{\mathcal{G},\theta}} (\theta - \theta_s). \end{aligned} \quad (29)$$

463 Since $\theta_{\text{lin},\epsilon}^*$ is the optimal parameter minimizing the objective, the derivative of the objective with
 464 respect to θ at $\theta = \theta_{\text{lin},\epsilon}^*$ is zero:

$$\begin{aligned} 0 &= -\frac{1}{N} \sum_{v \in \mathcal{V}_{\text{train}}} \mathbf{J}_{h_v^{\mathcal{G},\theta}}^{\top} \nabla_{h_v^{\mathcal{G},\theta}}^2 \mathcal{L}(h_v^{\mathcal{G},\theta_s}) \mathbf{J}_{h_v^{\mathcal{G},\theta}} (\theta_{\text{lin},\epsilon}^* - \theta_s) + \lambda (\theta_{\text{lin},\epsilon}^* - \theta_s) \\ &\quad + \sum_{v \in \mathcal{V}_{\text{train}}} \epsilon \left(\nabla_{\theta} \mathcal{L}(h_v^{\mathcal{G},\theta_s}) - \nabla_{\theta} \mathcal{L}(h_v^{\mathcal{G}^{-\frac{1}{N}},\theta_s}) \right). \end{aligned} \quad (30)$$

465 Rearranging the terms, we obtain:

$$\theta_{\text{lin},\epsilon}^* = \theta_s + \epsilon \mathbf{G}^{-1} \sum_{v \in \mathcal{V}_{\text{train}}} \left(\nabla_{\theta} \mathcal{L}(h_v^{\mathcal{G},\theta_s}) - \nabla_{\theta} \mathcal{L}(h_v^{\mathcal{G}^{-\frac{1}{N}},\theta_s}) \right), \quad (31)$$

466 where $\mathbf{G} := \frac{1}{N} \sum_{v \in \mathcal{V}_{\text{train}}} \mathbf{J}_{h_v^{\mathcal{G},\theta}}^{\top} \nabla_{h_v^{\mathcal{G},\theta}}^2 \mathcal{L}(h_v^{\mathcal{G},\theta_s}) \mathbf{J}_{h_v^{\mathcal{G},\theta}} + \lambda \mathbf{I}$ is the generalized Gauss-Newton matrix.

467 Taking the derivative with respect to ϵ , we obtain:

$$\frac{\partial \theta_{\text{lin},\epsilon}^*}{\partial \epsilon} = \lim_{\epsilon \rightarrow 0} \frac{\theta_{\text{lin},\epsilon}^* - \theta_s}{\epsilon} = \mathbf{G}^{-1} \sum_{v \in \mathcal{V}_{\text{train}}} \left(\nabla_{\theta} \mathcal{L}(h_v^{\mathcal{G},\theta_s}) - \nabla_{\theta} \mathcal{L}(h_v^{\mathcal{G}^{-\frac{1}{N}},\theta_s}) \right), \quad (32)$$

468 which confirms that the linearized edge-edit PBRF objective yields the same influence function as
 469 derived in Equation (10).

C Description of LiSSA

To approximate the inverse of the generalized Gauss–Newton Hessian-vector product $\mathbf{G}^{-1}v$, we employ the LiSSA algorithm [1]. LiSSA estimates $\mathbf{G}^{-1}v$ by iteratively accumulating powers of the residual matrix $(\mathbf{I} - \mathbf{G})$ applied to the vector v . When the spectral radius of $(\mathbf{I} - \mathbf{G})$ is less than 1, the inverse can be expressed using the Neumann series:

$$\mathbf{G}^{-1}v = \sum_{k=0}^{\infty} (\mathbf{I} - \mathbf{G})^k v. \quad (33)$$

Letting $r^{(K)} = \sum_{k=0}^K (\mathbf{I} - \mathbf{G})^k v$, the iteration is defined recursively as:

$$r^{(0)} = v, \quad r^{(k+1)} = v + (\mathbf{I} - \mathbf{G})r^{(k)}. \quad (34)$$

In practice, we perform the update in Equation (34) until convergence. The iteration is terminated early if the update difference $\|r^{(k+1)} - r^{(k)}\|$ falls below a predefined threshold, or when the number of iterations reaches 10,000.

Since the spectral radius of $(\mathbf{I} - \mathbf{G})$ is not necessarily less than 1, we rescale the matrix to ensure convergence. Specifically, we define a scaled matrix $\mathbf{G}_s = \frac{1}{s}\mathbf{G}$, where $s > \lambda_{\max}(\mathbf{G})$, so that the spectral radius of $(\mathbf{I} - \mathbf{G}_s)$ is less than 1. The inverse is then computed via

$$\mathbf{G}^{-1}v = \frac{1}{s}\mathbf{G}_s^{-1}v,$$

and LiSSA is applied to approximate $\mathbf{G}_s^{-1}v$.

To avoid explicitly storing the generalized Gauss–Newton matrix \mathbf{G} , the matrix-vector product $\mathbf{G}r^{(k)}$ in Equation (34) is computed approximately using a Jacobian-based heuristic. Specifically, we compute the Jacobian-vector product $x = \mathbf{J}_{h\theta_s} r^{(k)}$, apply the Hessian to obtain $x \leftarrow \mathbf{H}_{h_s} x$, and finally compute the transposed Jacobian-vector product $\mathbf{G}r^{(k)} = \mathbf{J}_{h\theta_s}^\top x$.

D Experimental result on other datasets and GNNs

In this section, we present scatter plots for additional non-convex GNNs and datasets. Figure 6 shows the results for additional datasets, while Figure 7 and Figure 8 show the results for ChebNet [12] and GAT [31], respectively. Under these settings, our influence function accurately predicts the actual influence, consistently showing a correlation above 0.8.

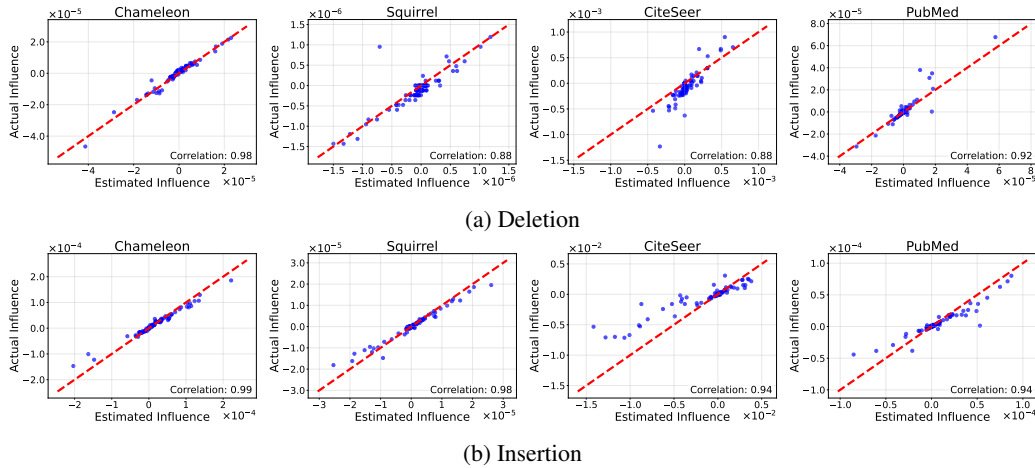


Figure 6: The scatter plot of predicted influence and actual influence on four-layer GCN. The x-axis represents the predicted influence and y-axis represents the actual influence, and the red-dotted line represents the perfect alignment.

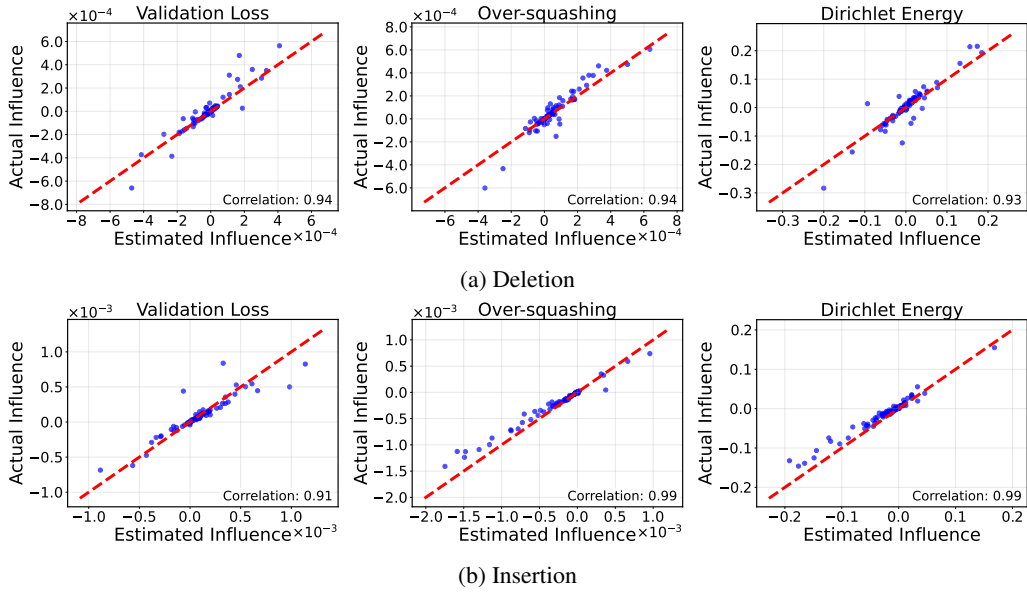


Figure 7: The scatter plot of predicted influence and actual influence on two-layer ChebNet. The x-axis represents the predicted influence and y-axis represents the actual influence, and the red-dotted line represents the perfect alignment.

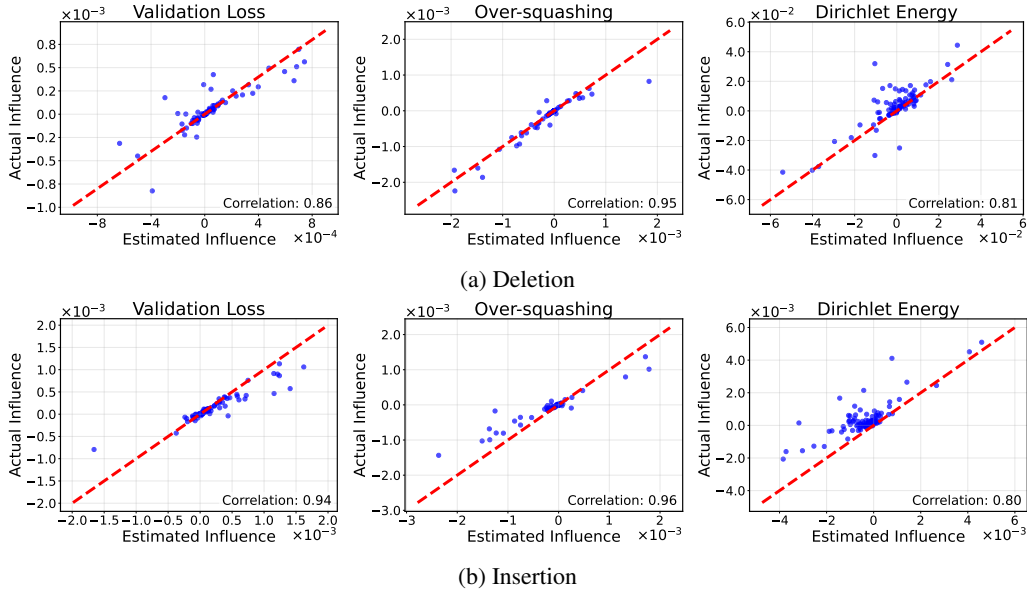
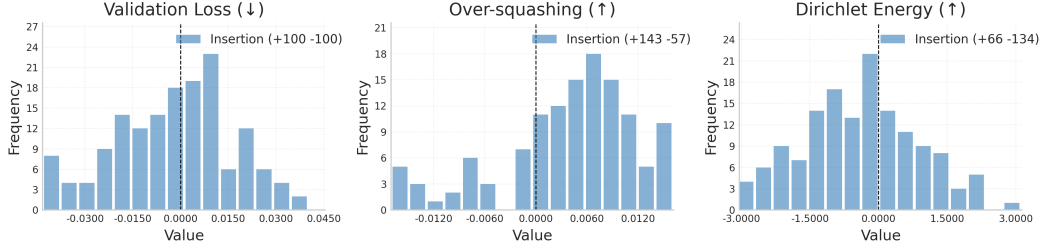


Figure 8: The scatter plot of predicted influence and actual influence on two-layer GAT. The x-axis represents the predicted influence and y-axis represents the actual influence, and the red-dotted line represents the perfect alignment.



(a) FoSR [18]

Figure 9: Histograms of the estimated influence of edge insertions selected by FoSR [18], evaluated on a four-layer GCN trained on the Texas dataset. Influences are measured across validation loss, over-squashing, and over-smoothing. Arrows (\downarrow / \uparrow) indicate the desired direction of each measurement (decrease/increase).

E Analysis on FoSR

In this section, we analyze the edge rewiring strategy of FoSR [18] using our influence function. FoSR rewires the graph by inserting edges that increase the spectral gap. Figure 9 presents the influence of the inserted edges in terms of validation loss, over-squashing, and Dirichlet energy.

Among the 200 edges inserted by FoSR, 143 are predicted to reduce over-squashing, suggesting that FoSR effectively identifies and adds edges that mitigate this issue. However, a similar limitation to that observed in BORF [24] emerges: the inserted edges do not necessarily improve the validation loss and may even exacerbate over-smoothing. These results consistently show that mitigating over-squashing does not necessarily improve other evaluation metrics, highlighting the necessity of considering multiple perspectives.

F Experimental configuration

All experiments are conducted using NVIDIA GeForce RTX 3090, NVIDIA RTX A5000, and NVIDIA RTX A6000 GPUs. The experiments presented in the main text employ a 4-layer GCN model as a representative non-convex GNN. To produce the results in Table 1 and Table 2, we tune the model and training hyperparameters of a vanilla GCN over the following search space: learning rates $\{0.1, 0.03, 0.01\}$, hidden dimensions $\{32, 64\}$, and weight decays $\{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}, 10^{-7}\}$. Training is performed for 2000 epochs using the SGD optimizer. For influence function computation, we run the LiSSA algorithm for 10,000 iterations. The damping parameter λ is selected from $\{0.1, 0.01, 0.001, 0.0001\}$. We randomly sample 10,000 candidate edges for both deletion and insertion, and estimate their influence. For Table 1, the number of edges to edit is determined based on validation accuracy. The experiments are repeated for 10 independent runs using different random seeds provided by the BernNet implementation.

BORF [24] and **FoSR** [18], used in the analysis of edge rewiring methods, are applied with the default settings from the original paper. The number of rewired edges reported in Figure 4 is aggregated over 10 runs, with a total of 200 edges inserted. Unlike FoSR, which performs only edge insertions, BORF also considers edge deletions. To improve visualization, we reduce the number of edge deletions, as the influence scores for insertions exhibit a long-tailed distribution, making it difficult to display them on the same scale as deletions. Accordingly, edge deletions are performed with 120, 100, and 100 edges, respectively.

For the adversarial attack experiments in Table 2, **DICE** [32] and **PRBCD** [13] are implemented by modifying the PyTorch-based DeepRobust library [22], while maintaining its default settings. For our method, we consider both edge insertions and deletions, and select the operations with the highest influence scores for validation loss, and the lowest influence scores for over-squashing and Dirichlet energy.