# Paper Reading

**Terence Wang**

**2024/02/29**

## Contents

# Understanding Black-box Predictions via Influence Functions

This paper is what I am most interested in. Detailed summary is as follows.

This paper is mainly about utilizing **Influence Functions** to make models more explainable. To begin with, if we want to check out the influence of a single training point $z_i = (x_i, y_i)$ on the model's prediction, we can remove the training point and use the rest of training dataset to retrain the model, then we can compare the difference between the original loss and the new loss. Apparently, retraining means a lot of time and resources, so the paper proposes a method that uses **Influence Functions** to approximate the influence of a single training point. In fact, removing a single training point is just a special case of perturbing a training point, so the authors formulate the whole problem under the circumstance of perturbing a training point. However, it is not easy to compute $I_{up,loss}(z, z_{test}) = -\nabla_\theta L(z_{test}, \hat{\theta})^T H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})$. Therefore, the authors use implicit Hessian-vector products(HVPs) to efficiently approximate $s_{test} = H_{\hat{\theta}}^{-1} \nabla_\theta L(z_{test}, \hat{\theta})$ and then compute $I_{up,loss}(z, z_{test}) = -s_{test} \cdot \nabla_\theta L(z, \hat{\theta})$. This paper discusses two techniques for approximating $s_{test}$: **Conjugate gradients** and **Stochastic estimation**. Finally, several cases are listed to prove the effectiveness of **Influence Functions**: Understanding model behavior  Adversarial training examples  Debugging domain mismatch  Fixing mislabeled examples.