The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Sensitivity Analysis of Deep Neural Networks

Hai Shu
Department of Biostatistics
The University of Texas MD Anderson Cancer Center
Houston, Texas, USA

Hongtu Zhu
AI Labs, Didi Chuxing
Beijing, China
zhuhongtu@didiglobal.com

Abstract

Deep neural networks (DNNs) have achieved superior performance in various prediction tasks, but can be very vulnerable to adversarial examples or perturbations. Therefore, it is crucial to measure the sensitivity of DNNs to various forms of perturbations in real applications. We introduce a novel perturbation manifold and its associated influence measure to quantify the effects of various perturbations on DNN classifiers. Such perturbations include various external and internal perturbations to input samples and network parameters. The proposed measure is motivated by information geometry and provides desirable invariance properties. We demonstrate that our influence measure is useful for four model building tasks: detecting potential 'outliers', analyzing the sensitivity of model architectures, comparing network sensitivity between training and test sets, and locating vulnerable areas. Experiments show reasonably good performance of the proposed measure for the popular DNN models ResNet50 and DenseNet121 on CIFAR10 and MNIST datasets.

1 Introduction

Deep neural networks (DNNs) have exhibited impressive power in image classification and outperformed human detection in the ImageNet challenge (Russakovsky et al. 2015; He et al. 2015; 2016; Huang et al. 2017). Despite this huge success, it is well known that state-of-the-art DNNs can be sensitive to small perturbations (Szegedy et al. 2013; Goodfellow, Shlens, and Szegedy 2015; Moosavi-Dezfooli, Fawzi, and Frossard 2016; Carlini and Wagner 2017; Su, Vargas, and Kouichi 2017). This vulnerability has called into question their usage in safety-critical applications, including self-driving cars (Bojarski et al. 2016) and face recognition (Sharif et al. 2017), among many others (Akhtar and Mian 2018). There is rich literature on quantifying the sensitivity or robustness of DNNs to external perturbations that affect the input samples; see (Fawzi, Moosavi-Dezfooli, and Frossard 2017; Akhtar and Mian 2018; Novak et al. 2018). For instance, one popular robustness measure computes the minimum adversarial distortion for a given sample (Moosavi-Dezfooli, Fawzi, and Frossard 2016; Hein and Andriushchenko 2017; Weng et al. 2018). However, very little work has been done on measuring the effects of various internal perturbations to network trainable parameters on DNNs. To the best of our knowledge, (Cheney, Schrimpf, and Kreiman 2017) is the first paper to examine the robustness of AlexNet (Krizhevsky, Sutskever, and Hinton 2012) by tracking the classification performance over several chosen standard deviations of Gaussian perturbations to network weights.

The aim of this paper is to develop a novel perturbation manifold and its associated influence measure to evalute the effects of various perturbations to input samples and/or network trainable parameters. Our influence measure is a novel extension of the local influence measures proposed in (Zhu et al. 2007; Zhu, Ibrahim, and Tang 2011) to classification problems by using information geometry (Amari 1985; Amari and Nagaoka 2000). Compared with the existing methods (Akhtar and Mian 2018), we make the following two major methodological contributions.

Our influence measure is motivated by information geometry, and its calculation is computationally straightforward and does not require optimizing any objective function. When the dimension of the perturbation vector is larger than the number of classes, the perturbation manifold in (Zhu et al. 2007; Zhu, Ibrahim, and Tang 2011) has a singular metric tensor and thus fails to form a Riemannian manifold. We address this singularity issue by introducing a low-dimensional transform and show that our influence measure still provides the invariance under diffeomorphisms of the original perturbation. Such an invariance property is critical for assessing the simultaneous effects or comparing the individual impacts of different external and/or internal perturbations within or between DNNs without concerning their difference in scales, such as the comparison between perturbations to trainable parameters in a convolution layer and those in a batch normalization layer within a single DNN. In contrast, existing measures, such as the Jacobian norm (Novak et al. 2018) and Cook's local influence measure (Cook 1986), do not have this invariance property, leading to some misleading results.

Our proposed influence measure is applicable to various forms of external and internal perturbations and useful for four important model building tasks: (i) detecting potential 'outliers', (ii) analyzing the sensitivity of model architectures, (iii) comparing network sensitivity between training and test sets, and (iv) locating vulnerable areas. For task (i), downweighting outliers may be used to train a DNN with increased robustness. Task (ii) may serve as a guide to the improvement of an existing network architecture. Task (iii) can evaluate the heterogeneity of the model robustness between training and test sets, and combining tasks (i)–(iii) may be useful for selecting DNNs. For task (iv), the discovered vulnerable locations in a given image can be utilized to either craft adversarial examples or fortify a DNN with data augmentation. The application of our influence measure to tasks (i)-(iv) is illustrated for two popular DNNs, ResNet50 (He et al. 2016) and DenseNet121 (Huang et al. 2017), on the benchmark datasets CIFAR10 and MNIST.

2 Method

2.1 Perturbation Manifold

Given an input image $\boldsymbol{x}$ and a DNN model N with a trainable parameter vector $\boldsymbol{\theta}$, the prediction probability for the response variable $y \in \{1, \ldots, K\}$ is denoted as $P(y|\boldsymbol{x}, \boldsymbol{\theta}) = N_{\boldsymbol{\theta}}(y, \boldsymbol{x})$. Let $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_p)^T$ be a perturbation vector varying in an open subset $\Omega \subseteq \mathbb{R}^p$. The perturbation $\boldsymbol{\omega}$ can be flexibly imposed on $\boldsymbol{x}$, $\boldsymbol{\theta}$, or even the combination of $\boldsymbol{x}$ and $\boldsymbol{\theta}$. Denote $P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega})$ to be the prediction probability under perturbation $\boldsymbol{\omega}$ such that $\sum_{y=1}^{K} P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega}) = 1$. It is assumed that there is a $\boldsymbol{\omega}_0 \in \Omega$ such that $P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega}_0) = P(y|\boldsymbol{x}, \boldsymbol{\theta})$. Also, $\{P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega})\}_{y=1}^{K}$ is assumed to be positive and sufficiently smooth for all $\boldsymbol{\omega} \in \Omega$.

Following the development in (Zhu et al. 2007; Zhu, Ibrahim, and Tang 2011), we may define $\mathcal{M} = \{P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega}) : \boldsymbol{\omega} \in \Omega\}$ as a perturbation manifold. The tangent space of $\mathcal{M}$ at $\boldsymbol{\omega}$ is denoted by $T_{\boldsymbol{\omega}}$, which is spanned by $\{\partial \ell(\boldsymbol{\omega}|y, \boldsymbol{x}, \boldsymbol{\theta})/\partial \omega_i\}_{i=1}^{p}$, where $\ell(\boldsymbol{\omega}|y, \boldsymbol{x}, \boldsymbol{\theta}) = \log P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega})$. Let $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}) = \sum_{y=1}^{K} \partial_{\boldsymbol{\omega}}^T \ell(\boldsymbol{\omega}|y, \boldsymbol{x}, \boldsymbol{\theta}) \partial_{\boldsymbol{\omega}} \ell(\boldsymbol{\omega}|y, \boldsymbol{x}, \boldsymbol{\theta}) P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega})$ with $\partial_{\boldsymbol{\omega}} = (\partial/\partial\omega_1, \ldots, \partial/\partial\omega_p)$. If $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega})$ is positive definite, then for any two tangent vectors $v_i(\boldsymbol{\omega}) = \boldsymbol{h}_i^T \partial_{\boldsymbol{\omega}}^T \ell(\boldsymbol{\omega}|y, \boldsymbol{x}, \boldsymbol{\theta}) \in T_{\boldsymbol{\omega}}$, $i = 1, 2$, where $\boldsymbol{h}_i^T$ denotes the coordinate vector of $v_i(\boldsymbol{\omega})$ on the basis $\partial_{\boldsymbol{\omega}} \ell(\boldsymbol{\omega}|y, \boldsymbol{x}, \boldsymbol{\theta})$, the inner product can be defined by

$$\begin{aligned}\langle v_1(\boldsymbol{\omega}), v_2(\boldsymbol{\omega})\rangle &= \sum_{y=1}^{K} v_1(\boldsymbol{\omega}) v_2(\boldsymbol{\omega}) P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega}) \\ &= \boldsymbol{h}_1^T \mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}) \boldsymbol{h}_2. \qquad (1)\end{aligned}$$

Subsequently, the length of $v_1(\boldsymbol{\omega})$ is given by

$$\|v_1(\boldsymbol{\omega})\| = \sqrt{\langle v_1(\boldsymbol{\omega}), v_1(\boldsymbol{\omega})\rangle} = \left[\boldsymbol{h}_1^T \mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}) \boldsymbol{h}_1\right]^{1/2}.$$

With the above inner product defined by $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega})$, $\mathcal{M}$ is a Riemannian manifold and $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega})$ is the Riemannian metric tensor (Amari 1985; Amari and Nagaoka 2000).

We need the positive definiteness of $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega})$. However, $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega})$ as a sum of K rank-1 matrices has $\text{rank}(\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega})) \leq K$, so it is a singular matrix when $K < p$. The case $K < p$ is true in many classification problems since the number of classes is often much smaller than the dimension of $\boldsymbol{\omega}$. The singularity of $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega})$ indicates that the p tangent vectors $\partial \ell(\boldsymbol{\omega}|y, \boldsymbol{x}, \boldsymbol{\theta})/\partial \omega_i$ are linearly dependent and thus some components of $\boldsymbol{\omega}$ are redundant. In addition, our focus is on the small perturbations around $\boldsymbol{\omega}_0$. We hence transform the p-dimensional $\boldsymbol{\omega}$ to be a vector $\boldsymbol{\nu}$ such that $\mathbf{G}_{\boldsymbol{\nu}}(\boldsymbol{\nu})$ is positive definite in a small neighborhood of $\boldsymbol{\nu}_0$ that corresponds to $\boldsymbol{\omega}_0$.

Our low-dimensional transform is implemented as follows. We first obtain a compact singular value decomposition (cSVD) of $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0)$. For very large p, rather than the direct but extremely expensive cSVD computation of the $p \times p$ matrix, we apply a computationally efficient approach using the cSVD of the much smaller $p \times K$ matrix $\mathbf{L}_0 = [\partial_{\boldsymbol{\omega}}^T \ell(\boldsymbol{\omega}_0|y, \boldsymbol{x}, \boldsymbol{\theta}) P^{1/2}(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega}_0)]_{1 \leq y \leq K}$ by noticing that $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0) = \mathbf{L}_0 \mathbf{L}_0^T$. Let $r_0 = \text{rank}(\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0))$. The usual cSVD computation can easily yield that $\mathbf{L}_0 = \mathbf{B}_0 \mathbf{A}_0$ and $\mathbf{A}_0 \mathbf{A}_0^T = \mathbf{U}_A \boldsymbol{\Lambda}_0 \mathbf{U}_A^T$, where $\mathbf{B}_0$ is a $p \times r_0$ matrix with orthonormal columns, $\mathbf{U}_A$ is a $r_0 \times r_0$ orthogonal matrix and $\boldsymbol{\Lambda}_0$ is a $r_0 \times r_0$ diagonal matrix. We hence obtain the cSVD: $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0) = \mathbf{U}_0 \boldsymbol{\Lambda}_0 \mathbf{U}_0^T$ with $\mathbf{U}_0 = \mathbf{B}_0 \mathbf{U}_A$. Define the desirable transform of $\boldsymbol{\omega} \in \Omega$ by $\boldsymbol{\nu} = \boldsymbol{\Lambda}_0^{1/2} \mathbf{U}_0^T \boldsymbol{\omega}$. Denote $P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\nu}) = P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega} = \mathbf{U}_0 \boldsymbol{\Lambda}_0^{-1/2} \boldsymbol{\nu} + \boldsymbol{\xi}_0)$, where $\boldsymbol{\xi}_0 = \boldsymbol{\omega}_0 - \mathbf{U}_0 \boldsymbol{\Lambda}_0^{-1/2} \boldsymbol{\nu}_0$ and $\boldsymbol{\nu}_0 = \boldsymbol{\Lambda}_0^{1/2} \mathbf{U}_0^T \boldsymbol{\omega}_0$. It follows from $\partial_{\boldsymbol{\nu}} \ell = \partial_{\boldsymbol{\omega}} \ell \mathbf{U}_0 \boldsymbol{\Lambda}_0^{-1/2}$ that $\mathbf{G}_{\boldsymbol{\nu}}(\boldsymbol{\nu}_0) = \mathbf{I}$. By the smoothness of $P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega})$ in $\boldsymbol{\omega} \in \Omega$, the metric tensor $\mathbf{G}_{\boldsymbol{\nu}}(\boldsymbol{\nu})$ is positive definite in an open ball $B_{\boldsymbol{\nu}_0}$ centered at $\boldsymbol{\nu}_0$.

Definition 1. *We define the Riemannian manifold* $\mathcal{M}_{\boldsymbol{\nu}_0} = \{P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\nu}) : \boldsymbol{\nu} \in B_{\boldsymbol{\nu}_0}\}$ *with the inner product defined by* $\mathbf{G}_{\boldsymbol{\nu}}(\boldsymbol{\nu})$ *in* (1) *as the perturbation manifold around* $\boldsymbol{\nu}_0$.

2.2 Influence Measure

Let $f(\boldsymbol{\omega})$ be the objective function of interest for sensitivity analysis. We define the influence measure to evaluate the discrepancy of the objective function $f(\boldsymbol{\omega})$ at two points, $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$, corresponding to $\boldsymbol{\nu}_i = \boldsymbol{\Lambda}_0^{1/2} \mathbf{U}_0^T \boldsymbol{\omega}_i$, $i = 1, 2$ on the perturbation manifold $\mathcal{M}_{\boldsymbol{\nu}_0}$. Let $C(t) = P(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\nu}(t))$ be a smooth curve on $\mathcal{M}_{\boldsymbol{\nu}_0}$ connecting $\boldsymbol{\nu}_1 = \boldsymbol{\nu}(t_1)$ to $\boldsymbol{\nu}_2 = \boldsymbol{\nu}(t_2)$, where $\boldsymbol{\nu}(t) = \boldsymbol{\Lambda}_0^{1/2} \mathbf{U}_0^T \boldsymbol{\omega}(t)$ with a smooth curve $\boldsymbol{\omega}(t)$ connecting $\boldsymbol{\omega}_1 = \boldsymbol{\omega}(t_1)$ to $\boldsymbol{\omega}_2 = \boldsymbol{\omega}(t_2)$. The distance between $\boldsymbol{\nu}_1$ and $\boldsymbol{\nu}_2$ along the curve $C(t)$ is defined by

$$S_C(\boldsymbol{\nu}_1, \boldsymbol{\nu}_2) = \int_{t_1}^{t_2} \left[\dot{\boldsymbol{\nu}}(t)^T \mathbf{G}_{\boldsymbol{\nu}}(\boldsymbol{\nu}(t)) \dot{\boldsymbol{\nu}}(t)\right]^{1/2} dt,$$

with $\dot{\boldsymbol{\nu}}(t) = d\boldsymbol{\nu}(t)/dt$. Following (Zhu, Ibrahim, and Tang 2011), the influence measure for $f(\boldsymbol{\omega})$ along $C(t)$ is given by

$$\mathrm{I}_C(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) = \frac{[f(\boldsymbol{\omega}_1) - f(\boldsymbol{\omega}_2)]^2}{S_C^2(\boldsymbol{\nu}_1, \boldsymbol{\nu}_2)}.$$

Let $\boldsymbol{\omega}(0) = \boldsymbol{\omega}_0$, then $\boldsymbol{\nu}(0) = \boldsymbol{\nu}_0$. We define the (first-order) local influence measure of $f(\boldsymbol{\omega})$ at $\boldsymbol{\omega}_0$ by

$$\mathrm{FI}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0) = \max_{C} \lim_{t \to 0} \mathrm{I}_C(\boldsymbol{\omega}(t), \boldsymbol{\omega}(0)). \qquad (2)$$

Denote $\boldsymbol{h}_\nu = \dot{\boldsymbol{\nu}}(0) = \boldsymbol{\Lambda}_0^{1/2} \mathbf{U}_0^T \boldsymbol{h}_\omega$, $\boldsymbol{h}_\omega = \dot{\boldsymbol{\omega}}(0)$, $\dot{\boldsymbol{\omega}}(t) = d\boldsymbol{\omega}/dt$, $\nabla_{f(\boldsymbol{\omega}_0)} = \partial_{\boldsymbol{\omega}} f|_{\boldsymbol{\omega}=\boldsymbol{\omega}_0}$, and $\mathbf{H}_{f(\boldsymbol{\omega}_0)} = \frac{\partial^2 f}{\partial \boldsymbol{\omega} \partial \boldsymbol{\omega}^T}\big|_{\boldsymbol{\omega}=\boldsymbol{\omega}_0}$. Plugging $S_C^2(\boldsymbol{\nu}(t), \boldsymbol{\nu}(0)) = t^2 \boldsymbol{h}_\nu^T \mathbf{G}_{\boldsymbol{\nu}}(\boldsymbol{\nu}_0) \boldsymbol{h}_\nu + o(t^2)$ and $f(\boldsymbol{\omega}(t)) = f(\boldsymbol{\omega}(0)) + \nabla_{f(\boldsymbol{\omega}_0)} \boldsymbol{h}_\omega t + \frac{1}{2}(\boldsymbol{h}_\omega^T \mathbf{H}_{f(\boldsymbol{\omega}_0)} \boldsymbol{h}_\omega +$

$\nabla_{f(\boldsymbol{\omega}_0)}\frac{d^2\boldsymbol{\omega}(0)}{dt^2})t^2 + o(t^2)$ into (2) yields the closed form

$$\begin{aligned} \mathrm{FI}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0) &= \max_{\boldsymbol{h}_\nu} \frac{\boldsymbol{h}_\nu^T \nabla_{f(\boldsymbol{\nu}_0)}^T \nabla_{f(\boldsymbol{\nu}_0)} \boldsymbol{h}_\nu}{\boldsymbol{h}_\nu^T \mathbf{G}_{\boldsymbol{\nu}}(\boldsymbol{\nu}_0) \boldsymbol{h}_\nu} \\ &= \nabla_{f(\boldsymbol{\nu}_0)} \nabla_{f(\boldsymbol{\nu}_0)}^T \qquad (3) \\ &= \nabla_{f(\boldsymbol{\omega}_0)} \mathbf{G}_{\boldsymbol{\omega}}^{\dagger}(\boldsymbol{\omega}_0) \nabla_{f(\boldsymbol{\omega}_0)}^T, \end{aligned}$$

where $\nabla_{f(\boldsymbol{\nu}_0)} := \nabla_{f(\boldsymbol{\omega}_0)} \mathbf{U}_0 \boldsymbol{\Lambda}_0^{-1/2}$, $\mathbf{G}_{\boldsymbol{\omega}}^{\dagger}(\boldsymbol{\omega}_0)$ is the pseudoinverse of $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0)$, and we used the identities $\mathbf{G}_{\boldsymbol{\nu}}(\boldsymbol{\nu}_0) = \mathbf{I}$ and $\partial_{\boldsymbol{\omega}} f \partial_t \boldsymbol{\omega} = \partial_{\boldsymbol{\omega}} f \partial_{\boldsymbol{\nu}} \boldsymbol{\omega} \partial_t \boldsymbol{\nu} = \partial_{\boldsymbol{\omega}} f \partial_{\boldsymbol{\nu}} \boldsymbol{\omega} \partial_{\boldsymbol{\omega}} \boldsymbol{\nu} \mathbf{U}_0 \boldsymbol{\Lambda}_0^{-1/2} \partial_t \boldsymbol{\nu} = \partial_{\boldsymbol{\omega}} f \mathbf{U}_0 \boldsymbol{\Lambda}_0^{-1/2} \partial_t \boldsymbol{\nu}$.

Definition 2. *We define the influence measure of* $f(\boldsymbol{\omega})$ *at* $\boldsymbol{\omega}_0$ *by* $\mathrm{FI}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0)$ *given in* (2) *with the closed form in* (3).

Theorem 1. *Suppose that* $\boldsymbol{\varphi}$ *is a diffeomorphism of* $\boldsymbol{\omega}$. *Then,* $\mathrm{FI}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0)$ *is invariant with respect to any reparameterization corresponding to* $\boldsymbol{\varphi}$.

Proof. Let $\boldsymbol{\varphi} = \boldsymbol{\varphi}(\boldsymbol{\omega})$, $\boldsymbol{\omega} = \boldsymbol{\omega}(\boldsymbol{\varphi})$, and $\boldsymbol{\varphi}_0 = \boldsymbol{\varphi}(\boldsymbol{\omega}_0)$. Denote their Jacobian matrices by $\boldsymbol{\Phi} = \partial\boldsymbol{\varphi}/\partial\boldsymbol{\omega}$ and $\boldsymbol{\Psi} = \partial\boldsymbol{\omega}/\partial\boldsymbol{\varphi}$. Differentiating $\boldsymbol{\omega} = \boldsymbol{\omega}(\boldsymbol{\varphi}(\boldsymbol{\omega}))$ with respect to $\boldsymbol{\omega}$ yields $\mathbf{I} = \boldsymbol{\Psi}\boldsymbol{\Phi}$. Denote $\boldsymbol{\Psi}_0 = \boldsymbol{\Psi}(\boldsymbol{\varphi}_0)$, $\boldsymbol{\Phi}_0 = \boldsymbol{\Phi}(\boldsymbol{\omega}_0)$, $\dot{\boldsymbol{\omega}}_0 = \dot{\boldsymbol{\omega}}(0)$ and $\dot{\boldsymbol{\varphi}}_0 = d\boldsymbol{\varphi}(0)/dt$. We have

$$\begin{aligned} \mathrm{FI}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0) &= \max_{\boldsymbol{h}_\omega} \frac{\boldsymbol{h}_\omega^T \nabla_{f(\boldsymbol{\omega}_0)}^T \nabla_{f(\boldsymbol{\omega}_0)} \boldsymbol{h}_\omega}{\boldsymbol{h}_\omega^T \mathbf{U}_0 \boldsymbol{\Lambda}_0 \mathbf{U}_0^T \boldsymbol{h}_\omega} \\ &= \max_{\dot{\boldsymbol{\omega}}_0} \frac{\dot{\boldsymbol{\omega}}_0^T \nabla_{f(\boldsymbol{\omega}_0)}^T \nabla_{f(\boldsymbol{\omega}_0)} \dot{\boldsymbol{\omega}}_0}{\dot{\boldsymbol{\omega}}_0^T \mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0) \dot{\boldsymbol{\omega}}_0} \\ &= \max_{\dot{\boldsymbol{\omega}}_0} \frac{\dot{\boldsymbol{\omega}}_0^T \boldsymbol{\Phi}_0^T \boldsymbol{\Psi}_0^T \nabla_{f(\boldsymbol{\omega}_0)}^T \nabla_{f(\boldsymbol{\omega}_0)} \boldsymbol{\Psi}_0 \boldsymbol{\Phi}_0 \dot{\boldsymbol{\omega}}_0}{\dot{\boldsymbol{\omega}}_0^T \boldsymbol{\Phi}_0^T \boldsymbol{\Psi}_0^T \mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0) \boldsymbol{\Psi}_0 \boldsymbol{\Phi}_0 \dot{\boldsymbol{\omega}}_0} \\ &= \max_{\dot{\boldsymbol{\varphi}}_0} \frac{\dot{\boldsymbol{\varphi}}_0^T \nabla_{f(\boldsymbol{\varphi}_0)}^T \nabla_{f(\boldsymbol{\varphi}_0)} \dot{\boldsymbol{\varphi}}_0}{\dot{\boldsymbol{\varphi}}_0^T \mathbf{G}_{\boldsymbol{\varphi}}(\boldsymbol{\varphi}_0) \dot{\boldsymbol{\varphi}}_0} = \mathrm{FI}_{\boldsymbol{\varphi}}(\boldsymbol{\varphi}_0). \end{aligned}$$

□

Theorem 1 shows the invariance of $\mathrm{FI}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0)$ under any diffeomorphic (e.g., scaling) reparameterization of the original perturbation vector $\boldsymbol{\omega}$ rather than $\boldsymbol{\nu}$. This result is analogous to those in (Zhu et al. 2007; Zhu, Ibrahim, and Tang 2011), but we extend it to cases where the original perturbation model $\mathcal{M}$ with $\mathbf{G}_{\boldsymbol{\omega}}(\boldsymbol{\omega})$ is not a Riemannian manifold, especially when $K < p$.

The invariance property is not enjoyed by the widely used Jacobian norm (Novak et al. 2018) and Cook's local influence measure (Cook 1986). For example, consider the perturbation $\boldsymbol{\alpha} + \Delta\boldsymbol{\alpha}$, where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_p)^T$ is a subvector of $(\boldsymbol{x}^T, \boldsymbol{\theta}^T)^T$, and the scaling version $\boldsymbol{\alpha}' + \Delta\boldsymbol{\alpha}'$ with $\boldsymbol{\alpha}' = k\boldsymbol{\alpha}$. Let $(\boldsymbol{\omega}, \boldsymbol{\omega}_0) = (\Delta\boldsymbol{\alpha}, \mathbf{0})$ and its scaling counterpart $(\boldsymbol{\omega}', \boldsymbol{\omega}_0') = (\Delta\boldsymbol{\alpha}', \mathbf{0})$. We have that the Jacobian norm

$$\|\mathbf{J}(\boldsymbol{\alpha})\|_F = \Big[\sum_{i=1}^{p} \Big(\frac{\partial f}{\partial \alpha_i}\Big|_{\boldsymbol{\omega}=\boldsymbol{\omega}_0}\Big)^2\Big]^{1/2} = k\|\mathbf{J}(\boldsymbol{\alpha}')\|_F \qquad (4)$$

and the Cook's local influence measure

$$\begin{aligned} C_{\boldsymbol{\eta},\boldsymbol{\omega}} &= \frac{1}{(1 + \nabla_{f(\boldsymbol{\omega}_0)} \nabla_{f(\boldsymbol{\omega}_0)}^T)^{1/2}} \frac{\boldsymbol{\eta}^T \mathbf{H}_{f(\boldsymbol{\omega}_0)} \boldsymbol{\eta}}{\boldsymbol{\eta}^T (\mathbf{I} + \nabla_{f(\boldsymbol{\omega}_0)}^T \nabla_{f(\boldsymbol{\omega}_0)}) \boldsymbol{\eta}} \\ &\neq C_{\boldsymbol{\eta},\boldsymbol{\omega}'} = C_{k\boldsymbol{\eta},\boldsymbol{\omega}'} \qquad (5) \end{aligned}$$

with $\boldsymbol{\omega}(t) = \boldsymbol{\omega}_0 + t\boldsymbol{\eta}$ are not scaling-invariant. This is problematic especially when the scale heterogeneity exists between parameters to which the perturbations are imposed. For instance, in the simultaneous perturbations to both input image $\boldsymbol{x}$ and trainable network parameters $\boldsymbol{\theta}$, i.e., $\boldsymbol{\alpha} = (\boldsymbol{x}^T, \boldsymbol{\theta}^T)^T$, the contribution of $\Delta\boldsymbol{x}$ appears to be exaggerated if $\boldsymbol{x}$ is scaled with larger values than $\boldsymbol{\theta}$. Another example is the comparison between perturbations to trainable parameters (weights and bias) in a convolution layer and those (shift/scale parameters) in a batch normalization layer. There are no uniform criteria for the scaling because either rescaling to a unit norm or keeping on the original scales seems to have its own advantages. However, our influence measure evades this scaling issue by utilizing the metric tensor of the perturbation manifold rather than that of the usual Euclidean space.

2.3 Perturbation Examples

In this subsection, we illustrate how to compute the proposed influence measure for a trained DNN model $P(y|\boldsymbol{x}, \boldsymbol{\theta}) = N_{\boldsymbol{\theta}}(y, \boldsymbol{x})$. We consider the following commonly used perturbations to the input image $\boldsymbol{x}$ or the trainable parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_1^T, \ldots, \boldsymbol{\theta}_L^T)^T$, where $\boldsymbol{\theta}_l$ are the parameters in the l-th trainable network layer.

- Case 1: $\boldsymbol{x} + \Delta\boldsymbol{x}$;
- Case 2: $\boldsymbol{\theta} + \Delta\boldsymbol{\theta}$;
- Case 3: $\boldsymbol{\theta}_l + \Delta\boldsymbol{\theta}_l$.

All three cases can be written in a unified form $\boldsymbol{\alpha} + \Delta\boldsymbol{\alpha}$ with $\boldsymbol{\alpha} \in \{\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\theta}_l\}$. Let the perturbation vector $\boldsymbol{\omega} = \Delta\boldsymbol{\alpha}$ and $\boldsymbol{\omega}_0 = \mathbf{0}$. For the influence measure $\mathrm{FI}_{\boldsymbol{\omega}}(\boldsymbol{\omega}_0)$ in (3), we have

$$\nabla_{f(\boldsymbol{\nu}_0)} = (\partial_{\boldsymbol{\alpha}} f|_{\boldsymbol{\omega}=\boldsymbol{\omega}_0}) \mathbf{U}_0 \boldsymbol{\Lambda}_0^{-1/2}, \qquad (6)$$

where $\boldsymbol{\Lambda}_0$ and $\mathbf{U}_0$ are obtained starting from matrix $\mathbf{L}_0 = [\partial_{\boldsymbol{\omega}}^T \ell(\boldsymbol{\omega}_0|y, \boldsymbol{x}, \boldsymbol{\theta}) P^{1/2}(y|\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\omega}_0)]_{1 \le y \le K}$ through $\mathbf{L}_0 = \mathbf{B}_0 \mathbf{A}_0$, $\mathbf{A}_0 \mathbf{A}_0^T = \mathbf{U}_A \boldsymbol{\Lambda}_0 \mathbf{U}_A^T$ and $\mathbf{U}_0 = \mathbf{B}_0 \mathbf{U}_A$. The component $\partial_{\boldsymbol{\omega}} \ell(\boldsymbol{\omega}_0|y, \boldsymbol{x}, \boldsymbol{\theta})$ in $\mathbf{L}_0$ is now computed by

$$\partial_{\boldsymbol{\omega}} \ell(\boldsymbol{\omega}_0|y, \boldsymbol{x}, \boldsymbol{\theta}) = \partial_{\boldsymbol{\alpha}} \log P(y|\boldsymbol{x}, \boldsymbol{\theta}). \qquad (7)$$

The gradients $\partial_{\boldsymbol{\alpha}} f|_{\boldsymbol{\omega}=\boldsymbol{\omega}_0}$ and $\partial_{\boldsymbol{\alpha}} \log P(y|\boldsymbol{x}, \boldsymbol{\theta})$ can be calculated easily via backpropagation (Goodfellow et al. 2016) in deep learning libraries like TensorFlow (Abadi et al. 2016) and Pytorch (Paszke et al. 2017).

Next, we consider a specific DNN example under Case 3. Consider the following feedforward DNN architecture before the softmax layer:

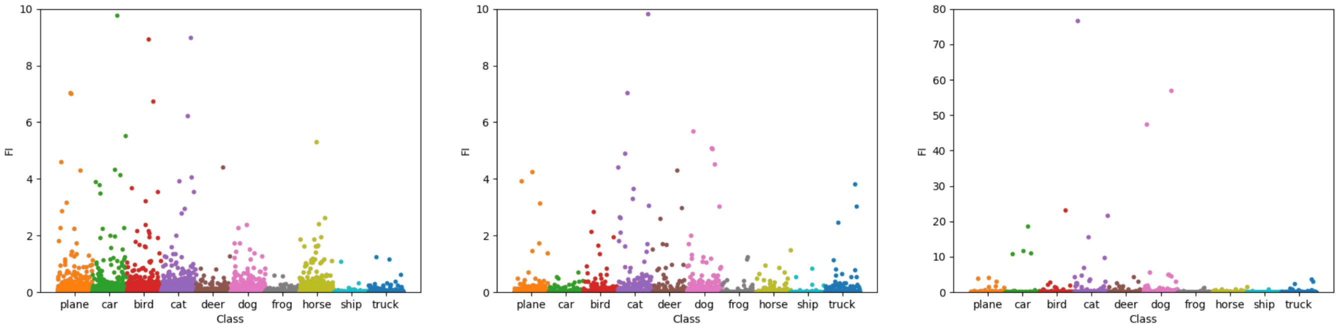
$$\boldsymbol{n}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\sigma}_L(\boldsymbol{\Theta}_L \boldsymbol{\sigma}_{L-1}(\cdots(\boldsymbol{\Theta}_3 \boldsymbol{\sigma}_2(\boldsymbol{\Theta}_2 \boldsymbol{\sigma}_1(\boldsymbol{\Theta}_1 \boldsymbol{x}))))) \in \mathbb{R}^K,$$

where $\boldsymbol{x} \in \mathbb{R}^{k_0}$, $\boldsymbol{\Theta}_l \in \mathbb{R}^{k_l \times k_{l-1}}$, $\boldsymbol{\theta}_l = \mathrm{vec}(\boldsymbol{\Theta}_l^T)$, and $\boldsymbol{\sigma}_l$'s are entry-wise activation functions. For notational simplicity, we set all bias terms to zero and consider the sigmoid function

$$\sigma(x) = [1 + \exp(-x)]^{-1} \text{ with } \dot{\sigma}(x) = \sigma(x)(1 - \sigma(x))$$

for all activation functions. Let $\boldsymbol{i}_l(\boldsymbol{x}, \boldsymbol{\theta})$ and $\boldsymbol{o}_l(\boldsymbol{x}, \boldsymbol{\theta})$ be the input and output vectors of the l-th layer such that $\boldsymbol{o}_l(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{\sigma}_l(\boldsymbol{i}_l(\boldsymbol{x}, \boldsymbol{\theta}))$ and $\boldsymbol{o}_0(\boldsymbol{x}, \boldsymbol{\theta}) = \boldsymbol{x}$. The softmax function is given by

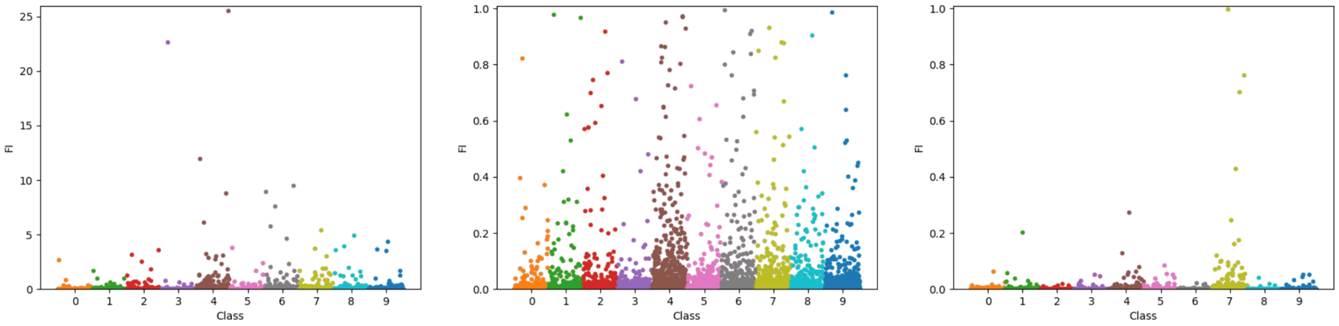
$$\boldsymbol{g}(\boldsymbol{z}) = \left(\frac{\exp(z_1)}{\sum_{k=1}^{K} \exp(z_k)}, \ldots, \frac{\exp(z_K)}{\sum_{k=1}^{K} \exp(z_k)}\right)^T.$$



(a) All FIs for ResNet50 on CIFAR10

(b) FIs≤10 for DenseNet121 on CIFAR10

(c) All FIs for DenseNet121 on CIFAR10



(d) All FIs for ResNet50 on MNIST

(e) FIs≤1 for ResNet50 on MNIST

(f) All FIs for DenseNet121 on MNIST

Figure 1: Manhattan plots for Setup 1.

The whole DNN model is

$$P(y|\boldsymbol{x},\boldsymbol{\theta}) = N_{\boldsymbol{\theta}}(y,\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{n}_{\boldsymbol{\theta}}(\boldsymbol{x}))_{[y]},$$

for $y = 1,\dots,K$, where $\boldsymbol{g}(\cdot)_{[y]}$ is the y-th entry of vector $\boldsymbol{g}(\cdot)$. Under Case 3, we have $\boldsymbol{\alpha} = \boldsymbol{\theta}_l$. Choose the objective function f to be the cross-entropy, i.e.,

$$f(\boldsymbol{\alpha},\boldsymbol{\omega}) = -\log P(y = y_{\text{true}}|\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\omega}).$$

Hence, in (6) we have $\partial_{\boldsymbol{\alpha}} f|_{\boldsymbol{\omega}=\boldsymbol{\omega}_0} = -\partial_{\boldsymbol{\alpha}}\log P(y = y_{\text{true}}|\boldsymbol{x},\boldsymbol{\theta})$. Then, to calculate the gradients in (6) and (7), we only need to consider $\partial_{\boldsymbol{\alpha}}\log P(y|\boldsymbol{x},\boldsymbol{\theta}) = \partial_{\boldsymbol{\theta}_l}\log(\boldsymbol{g}(\boldsymbol{n}_{\boldsymbol{\theta}}(\boldsymbol{x}))_{[y]})$. Note that $\partial_{\boldsymbol{z}}\log(\boldsymbol{g}(\boldsymbol{z})_{[y]}) = (\boldsymbol{e}_y - \boldsymbol{g}(\boldsymbol{z}))^T$, where $\boldsymbol{e}_y \in \mathbb{R}^K$ has 1 in the y-th entry and 0 in the others. Moreover, $\partial_{\boldsymbol{\theta}_l}\boldsymbol{n}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \mathbf{D}_L\boldsymbol{\Theta}_L\mathbf{D}_{L-1}\cdots\mathbf{D}_{l+1}\boldsymbol{\Theta}_{l+1}\mathbf{D}_l\mathbf{O}_{l-1}$, with $\mathbf{D}_l = \text{diag}(\{\dot{\sigma}(\boldsymbol{i}_l(\boldsymbol{x},\boldsymbol{\theta})_{[j]})\}_{j=1}^{k_l}) \in \mathbb{R}^{k_l\times k_l}$ and $\mathbf{O}_{l-1} = \text{diag}(\{\boldsymbol{o}_{l-1}^T(\boldsymbol{x},\boldsymbol{\theta}),\dots,\boldsymbol{o}_{l-1}^T(\boldsymbol{x},\boldsymbol{\theta})\}) \in \mathbb{R}^{k_l\times(k_{l-1}k_l)}$. Hence, for (6) and (7), we have

$$\begin{aligned}&\partial_{\boldsymbol{\alpha}}\log P(y|\boldsymbol{x},\boldsymbol{\theta})\\&= (\boldsymbol{e}_y - \boldsymbol{g}(\boldsymbol{n}_{\boldsymbol{\theta}}(\boldsymbol{x})))^T\mathbf{D}_L\boldsymbol{\Theta}_L\mathbf{D}_{L-1}\cdots\mathbf{D}_{l+1}\boldsymbol{\Theta}_{l+1}\mathbf{D}_l\mathbf{O}_{l-1}.\end{aligned}$$

3 Experiments

In this section, we investigate the performance of our local influence measure. We address the four tasks stated in Introduction through the following setups under the three perturbation cases in Section 2.3.

- Setup 1: Compute each training image's FI under Case 1, with f being the cross entropy, i.e., $f = -\log P(y = y_{\text{true}}|\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\omega})$.
- Setup 2: Let $f = -\log P(y = y_{\text{true}}|\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\omega})$.
  - Setup 2.1: Compute each training image's FI under Case 2.
  - Setup 2.2: Compute each trainable network layer's FI under Case 3 for each training image.
- Setup 3: Compute each image's FI under Case 1 for both training and test sets, where $f = -\log P(y = y_{\text{pred}}|\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\omega})$.
- Setup 4: Compute each pixel's FI under Case 1 for a given image. We adopt a multi-scale strategy taking into account the spatial effect. For each pixel, we set $\boldsymbol{x}$ in Case 1 to be the $k \times k$ square centered at the pixel with the scale $k \in \{1, 3, 5, 7\}$. We use $f = -\log P(y = y_{\text{pred}}|\boldsymbol{x},\boldsymbol{\theta},\boldsymbol{\omega})$.

For the cross-entropy like function f in Setups 3 and 4, we use the predicted label y_{pred} instead of the true label y_{true} for the prediction purpose rather than the training purpose in the first two setups. In Setup 4, the scale of the pixel-level FI is analogous to the convolutional kernel size.

We conduct experiments on the two benchmark datasets CIFAR10 and MNIST using the two popular DNN models ResNet50 (He et al. 2016) and DenseNet121 (Huang et al. 2017). Originally, there are 50,000 and 60,000 training images for CIFAR10 and MNIST, respectively. As the validation sets, we use randomly selected 10% of those images, with the same number for each class. No data augmentation is used for the training process. Both datasets have 10,000 test images. The prediction accuracy of our trained models is summarized in Table 1.

Table 1: Accuracy for models trained without data augmentation

	CIFAR10		MNIST	
Model	Training	Test	Training	Test
ResNet50	99.78%	88.70%	99.87%	99.29%
DenseNet121	99.87%	91.16%	99.998%	99.58%

3.1 Outlier Detection

We study the outlier detection ability of our proposed influence measure under Setup 1. Figure 1 illustrates the results of Setup 1 by using Manhattan plots. DenseNet121 generally has smaller FIs than ResNet50 for the two benchmark datasets, excluding several large FIs over 10 shown in Figure 1(c) for CIFAR10. The images with the top 5 largest FIs are displayed in Figure 2 for each case. Most of the 20 images, especially those in MNIST, are difficult even for human visual detection. This indicates the strong power of our influence measure in detecting outlier images.

We further examine the outlier detection power of our proposed influence measure by simulating outlier images from MNIST. Each outlier image was generated by overlapping two training digits of different classes that are shifted up to 4 pixels in each direction, with the true label randomly set to be one of the two classes. The two DNN models in Table 1 are trained with additional 50 epochs after incorporating 2700 and 300 simulated outlier images into the training and validation sets, with accuracies reduced up to 0.38% and 0.11% for respective training and testing. The original 54,000 training images are all treated as non-outliers. We compare the proposed FI measure with the Jacobian norm given in (4) using the cross-entropy as the objective function f. The maximal Cook's local influence, $\max_{\boldsymbol{\eta}} C_{\boldsymbol{\eta},\boldsymbol{\omega}}$, is not considered here due to the expensive computation of the very large Hessian matrix; see (5). Figure 3 shows the outlier detection results of the two considered measures. Although the receiver operating characteristic (ROC) curves of the two measures are almost overlapping, our FI measure significantly outperforms Jacobian norm in terms of the precision-recall (PR) curves that are more useful for highly unbalanced data (Davis and Goadrich 2006).

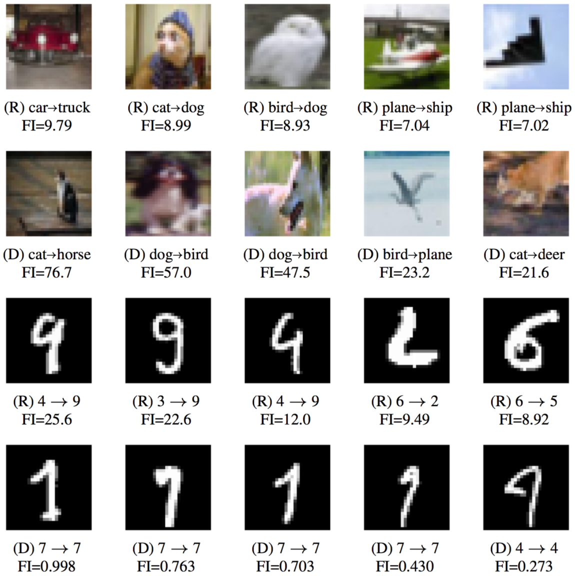
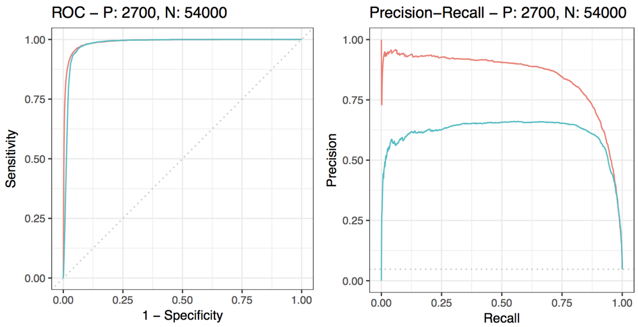
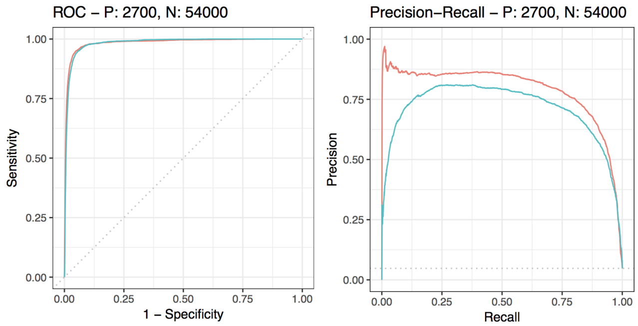

(R) car→truck FI=9.79
(R) cat→dog FI=8.99
(R) bird→dog FI=8.93
(R) plane→ship FI=7.04
(R) plane→ship FI=7.02
(D) cat→horse FI=76.7
(D) dog→bird FI=57.0
(D) dog→bird FI=47.5
(D) bird→plane FI=23.2
(D) cat→deer FI=21.6
(R) 4 → 9 FI=25.6
(R) 3 → 9 FI=22.6
(R) 4 → 9 FI=12.0
(R) 6 → 2 FI=9.49
(R) 6 → 5 FI=8.92
(D) 7 → 7 FI=0.998
(D) 7 → 7 FI=0.763
(D) 7 → 7 FI=0.703
(D) 7 → 7 FI=0.430
(D) 4 → 4 FI=0.273

Figure 2: Images with top 5 largest FIs in Setup 1 for ResNet50 (R) and DensetNet121 (D). Each subcaption shows $y_{\text{true}} \rightarrow y_{\text{pred}}$ and FI.



(a) ResNet50



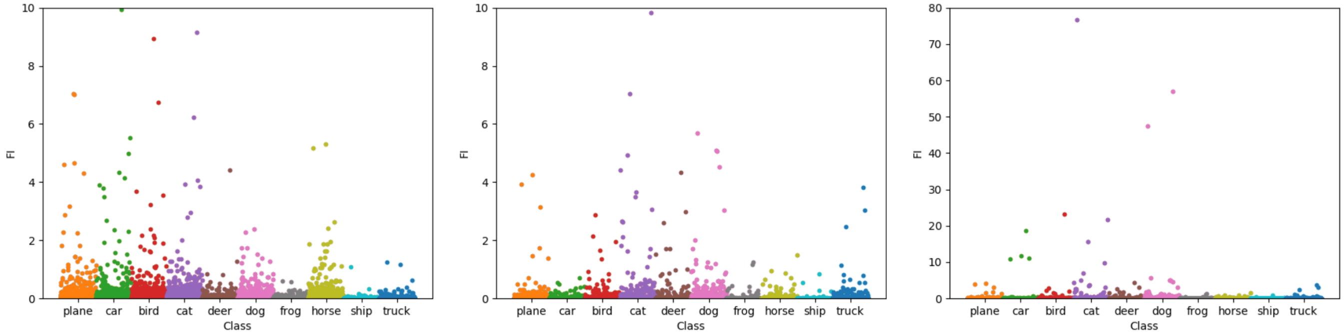
(b) DenseNet121

Figure 3: ROC and PR curves of our proposed FI measure (red) and the Jacobian norm (blue) on MNIST with simulated outliers.

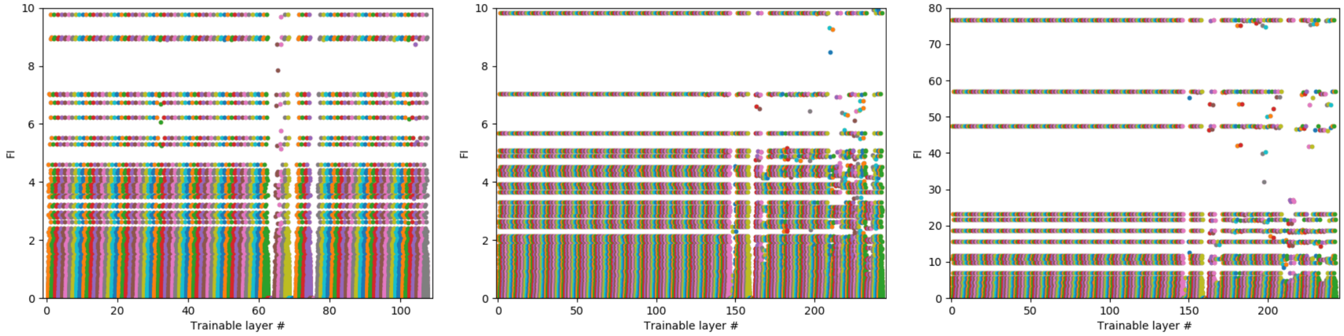
3.2 Sensitivity Analysis on DNN Architectures

We conduct the sensitivity analysis on DNN architectures under Setup 2. The invariance property of our FI measure shown in Theorem 1 enables us to fairly compare the effects of small perturbations to model parameters of different scales within or between DNNs. Setup 2.1 compares the sensitivity between the two DNNs, while Setup 2.2 undertakes the comparison across trainable layers within each single DNN.

The Manhattan plots for Setup 2 on CIFAR10 are presented in Figure 4; results for MNIST are provided in the Supplementary Material. The patterns on CIFAR10 under Setup 2.1 in Figure 4(a)-(c), with mostly smaller FIs for DensetNet121, are quite similar to those for Setup 1 in Figure 1(a)-(c), in-

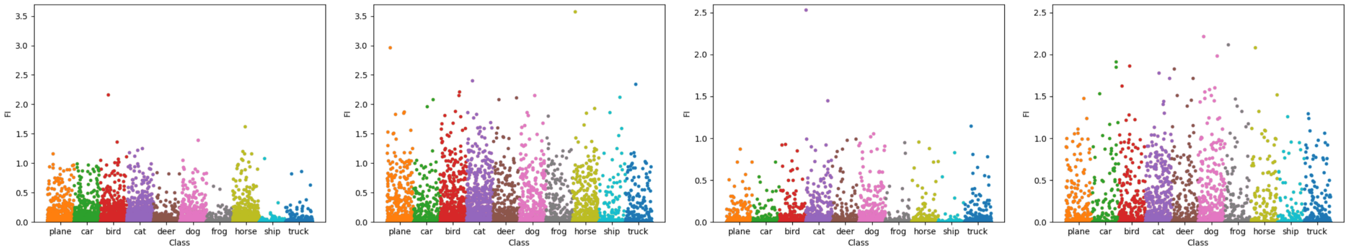


(a) All FIs for ResNet50 in Setup 2.1 (b) FIs≤10 for DenseNet121 in Setup 2.1 (c) All FIs for DenseNet121 in Setup 2.1



(d) All FIs for ResNet50 in Setup 2.2 (e) FIs≤10 for DenseNet121 in Setup 2.2 (f) All FIs for DenseNet121 in Setup 2.2

Figure 4: Manhattan plots for Setup 2 on CIFAR10.



(a) ResNet50 on CIFAR10 training set (b) ResNet50 on CIFAR10 test set (c) DenseNet121 on CIFAR10 training set (d) DenseNet121 on CIFAR10 test set

Figure 5: Manhattan plots for Setup 3 on CIFAR10.

dicating that DenseNet121 is generally less sensitive than ResNet50 to the infinitesimal perturbations to all network trainable parameters. From Figure 4(d)-(f) for Setup 2.2, we see stable patterns of FIs over the trainable layers for the two DNNs. Modifying their network architectures does not appear to be necessary here. Note that the FI value for the trainable parameters in each single network layer is theoretically dominated by that for all trainable parameters of the entire network, which is well supported by the comparison between Figure 4(a)-(c) and (d)-(f).

3.3 Sensitivity Comparison between Training and Test Sets

We compare the network sensitivity between training and test sets under Setup 3. Figure 5 and Table 2 show the FI values for Setup 3 on CIFAR10; results for MNIST are also provided in the Supplementary Material. In the figure and table, the test set has more slightly large FIs than the training set for both DNNs, while FIs are generally smaller in both sets for DenseNet121. We suggest to select a DNN model with similar sensitivity performance and smaller FI values on both training and test sets. Together with the results of Sections 3.1 and 3.2 shown in Figure 1 (a)-(c) and Figure 4 (a)-(c), and also with the model accuracies in Table 1, DenseNet121 is preferred over ResNet50 on CIFAR10 in terms of both sensitivity and accuracy.

3.4 Vulnerable Region Detection

We apply the multi-scale strategy in Setup 4 to detect the areas in an image that are vulnerable to small perturbations.

For Setup 4, the test images from the two benchmark datasets with the largest FI in Setup 3 by DenseNet121 are

Table 2: Percentiles of FI values for Setup 3 on CIFAR10

	ResNet50		DenseNet121	
Percentile	Training	Test	Training	Test
75th	2.87e-3	0.031	1.10e-4	1.83e-3
80th	5.38e-3	0.073	2.42e-4	6.57e-3
85th	0.010	0.160	6.00e-4	0.025
90th	0.023	0.343	1.78e-3	0.097
95th	0.064	0.678	7.80e-3	0.352
98th	0.177	0.999	0.037	0.755
99th	0.316	1.215	0.099	0.951
100th	2.160	3.579	2.533	2.215

illustrated in Figure 6. The vulnerable areas for both images are mainly in or around the object, and the image boundaries are generally less sensitive to perturbations. The figure also reasonably shows that the vulnerable areas expand as the scale of pixel-level FI increases.

Figure 7 illustrates the one-pixel adversarial attacks based on pixel-wise FI maps. The two selected test images are correctly predicted by ResNet50 with a high probability and also with a large FI in Setup 3. The pixel-wise FI map denotes the scale-1 pixel-level FI map for the MNIST image, and is the average scale-1 map over the three RGB channels for the CIFAR10 image. For each image, the attacked pixel is the one with the largest value in the pixel-wise FI map. We see that the prediction result significantly changes after slightly altering the selected pixel's value. This indicates that our FI measure is useful for discovering vulnerable locations and crafting adversarial examples.

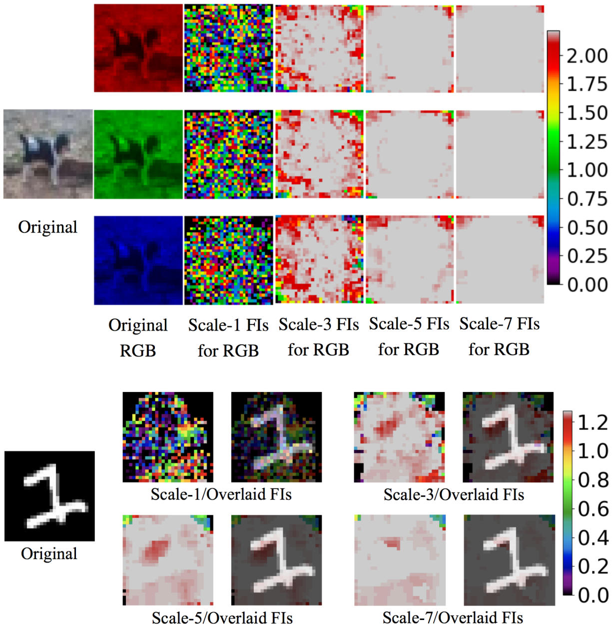
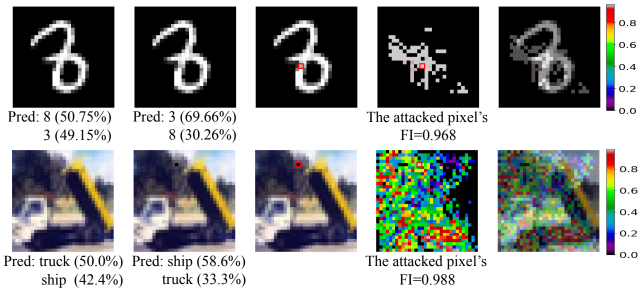


Figure 6: Multi-scale pixel-level FI maps for Setup 4 using DenseNet121. Results are shown for the test image with the largest FI in Setup 3. The CIFAR10 test image has Setup-3 FI $= 2.22$, $y_{\text{true}} =$ dog, and $y_{\text{pred}} =$ bird. The MNIST test image has Setup-3 FI $= 1.28$, $y_{\text{true}} = 1$, and $y_{\text{pred}} = 7$.



(a) Original images (b) Adversarial images (c) Annotated Adv. images (d) Pixel-wise FI maps (e) Overlaid FI maps

Figure 7: One-pixel adversarial attacks on ResNet50 using pixel-wise FI maps. The original images have $y_{\text{true}} = 8$ and $y_{\text{true}} =$ truck, respectively. The prediction probabilities from ResNet50 are given in the parentheses. The attacked pixels are framed in red.

4 Conclusion

In this paper, we introduced a novel perturbation manifold and its associated influence measure for sensitivity analysis of DNN classifiers. This new measure is constructed from a Riemannian manifold and provides the invariance property under any diffeomorphic (e.g., scaling) reparameterization of perturbations. This invariance property is not owned by the widely used measures like the Jacobian norm and Cook's local influence. Our influence measure performs very well for ResNet50 and DenseNet121 trained on CIFAR10 and MNIST datasets in the tasks of outlier detection, sensitivity comparison between network architectures and that between training and test sets, and vulnerable region detection.

References

Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; Kudlur, M.; Levenberg, J.; Monga, R.; Moore, S.; Murray, D. G.; Steiner, B.; Tucker, P.; Vasudevan, V.; Warden, P.; Wicke, M.; Yu, Y.; and Zheng, X. 2016. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 265–283.

Akhtar, N., and Mian, A. 2018. Threat of adversarial attacks on deep learning in computer vision: A survey. *arXiv preprint* arXiv:1801.00553.

Amari, S., and Nagaoka, H. 2000. *Methods of Information Geometry*. American Mathematical Society, Providence, RI.

Amari, S. 1985. *Differential-geometrical Methods in Statistics*. Springer-Verlag, New York.

Bojarski, M.; Del Testa, D.; Dworakowski, D.; Firner, B.; Flepp, B.; Goyal, P.; Jackel, L. D.; Monfort, M.; Muller, U.; Zhang, J.; Zhang, X.; Zhao, J.; and Zieba, K. 2016.

End to end learning for self-driving cars. *arXiv preprint* arXiv:1604.07316.

Carlini, N., and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy*, 39–57.

Cheney, N.; Schrimpf, M.; and Kreiman, G. 2017. On the robustness of convolutional neural networks to internal architecture and weight perturbations. *arXiv preprint* arXiv:1703.08245.

Cook, R. D. 1986. Assessment of local influence. *Journal of the Royal Statistical Society. Series B (Methodological)* 48(2):133–169.

Davis, J., and Goadrich, M. 2006. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine learning*, 233–240.

Fawzi, A.; Moosavi-Dezfooli, S.-M.; and Frossard, P. 2017. The robustness of deep networks: A geometrical perspective. *IEEE Signal Processing Magazine* 34(6):50–62.

Goodfellow, I.; Bengio, Y.; Courville, A.; and Bengio, Y. 2016. *Deep Learning*. Cambridge: MIT Press.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*. arXiv:1412.6572.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, 1026–1034.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Hein, M., and Andriushchenko, M. 2017. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, 2266–2276.

Huang, G.; Liu, Z.; Weinberger, K. Q.; and van der Maaten, L. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4700–4708.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 1097–1105.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2574–2582.

Novak, R.; Bahri, Y.; Abolafia, D. A.; Pennington, J.; and Sohl-Dickstein, J. 2018. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*. arXiv:1802.08760.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch. In *NIPS 2017 Autodiff Workshop*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Li, F.-F. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 115(3):211–252.

Sharif, M.; Bhagavatula, S.; Bauer, L.; and Reiter, M. K. 2017. Adversarial generative nets: Neural network attacks on state-of-the-art face recognition. *arXiv preprint* arXiv:1801.00349.

Su, J.; Vargas, D. V.; and Kouichi, S. 2017. One pixel attack for fooling deep neural networks. *arXiv preprint* arXiv:1710.08864.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. In *International Conference on Learning Representations*. arXiv:1312.6199.

Weng, T.-W.; Zhang, H.; Chen, P.-Y.; Yi, J.; Su, D.; Gao, Y.; Hsieh, C.-J.; and Daniel, L. 2018. Evaluating the robustness of neural networks: An extreme value theory approach. In *International Conference on Learning Representations*. arXiv:1801.10578.

Zhu, H.; Ibrahim, J. G.; Lee, S.; and Zhang, H. 2007. Perturbation selection and influence measures in local influence analysis. *The Annals of Statistics* 35(6):2565–2588.

Zhu, H.; Ibrahim, J. G.; and Tang, N. 2011. Bayesian influence analysis: a geometric approach. *Biometrika* 98(2):307–323.