Faculty of
Mathematics
and Informatics

VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
MODELLING AND DATA ANALYSIS
MASTER'S STUDY PROGRAMME

Influence functions - comparison of existing methods

Master's thesis

Author: Austėja Balkytė
VU email address: austeja.balkyte@mif.stud.vu.lt
Supervisor: Assoc. Prof. Dr. Viktor Skorniakov

Vilnius

2022

Abstract

In order to solve the problem of explainability of machine learning (ML) models, many solutions have been suggested in the recent years. One of them is a technique from robust statistics - influence functions (IF). However, many papers indicate that IF return outliers and mislabelled data as the most influential points from training set. In order to identify relevant examples that are more understandable for end users, alternative method was suggested: relative influence functions (RelatIF). In this paper, we compare how this method works on models where influence functions fail.

Keywords: Influence functions, RelatIF, explainability

Contents

1. Introduction

The growth of available data (Figure 1), less expensive data storage and more powerful processing has influenced the growth of machine learning applications. The more the need to identify profitable opportunities or avoid unknown risks is growing, the more sophisticated models are created. Applications in diverse fields, an influence in decision making and ability to solve problems in real-time systems (Ahmad and Chen (2020),Wang et al. (2020)) made a significant impact on creation of black box models.

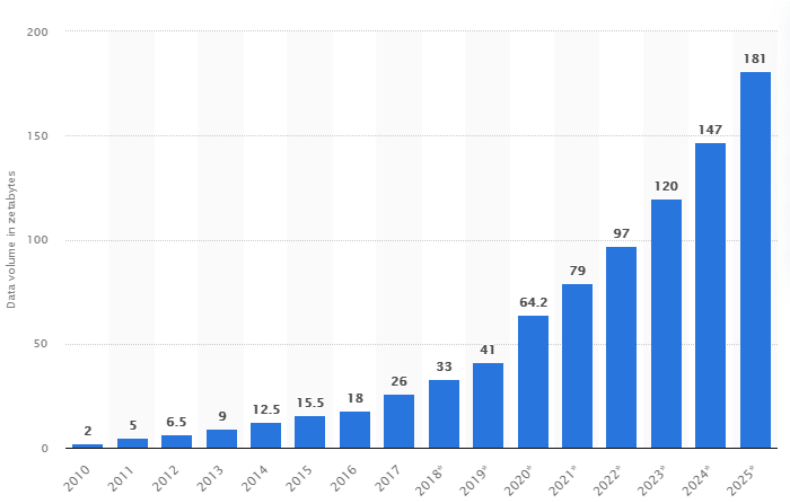

Figure 1: The total amount of data created, captured, copied, and consumed globally in recent years and forecast for upcoming years. Data from www.statista.com

Predictions made with complicated models are hard to explain and they might lead to unfair and wrong decisions, causing problems, such as biased predictions in criminal justice or wrong models in healthcare (Guidotti et al. (2018)). While the formal definition for explainability has yet to be defined, legal requirements for models applied in administration and justice are already in use (Bibal et al. (2021)). Therefore, it becomes really important to be able to explain why certain conclusions are drawn.

One of the ways to deal with black box models was proposed by Koh and Liang (2017) – influence functions (IF). IF demonstrate the effect of removing an individual training point on the model predictions at the test-time. It has been observed that IF return outliers and mislabelled training points as the most influential. Outliers' detection is an important part of data quality. As most of the real-world tasks have data that drifts over time (i.e. has a non-stationary environment), it is important to monitor quality of data.

The ability to detect outliers is important for machine learning engineers as it helps to debug the model itself. However, as highlighted by Bhatt et al. (2020), there is still a need of explainability techniques for end users. In order to deal with this problem, an alternative method has evolved – relative influence functions (RelatIF). They help to identify examples from training set that are more intuitive and understandable for end users.

However, it was indicated (Basu et al. (2020a)) that in non-convex environment influence functions do not work properly. In this paper, we compare RelatIF with IF. We conduct experiments in order to check whether RelatIF return more intuitive influential points and to see the impact of loss functions used for models. Models that are used have been chosen based on IF performance. Firstly, we analyse logistic regression where exact Hessian can be computed, then we compare Convolutional Neural Network (CNN) in binary classification problem and finally we analyse the results produced by three different CNN models for classification problems that vary in architecture, depth and width.

Paper is organized as follows. At first some preliminary information is introduced. Then literature review is given. In the fourth section influence functions are presented. In the fifth section we demonstrate why influence functions do not work on specific ML algorithms. In the sixth section, we introduce RelatIF. The seventh section is devoted to experimental findings. And finally, the conclusions are presented.

2. Preliminaries

- **Cross entropy** is a quantity, commonly used to evaluate discrepancy between two probability distributions. In ML, it is a measure of error for categorical multi-class classification problems, usually used to describe loss functions.

- Let X be an $n-$dimensional random vector with a density $p_\theta(x), x \in \mathbb{R}^n$ which depends on a parameter $\theta \in \Theta \subset \mathbb{R}^k$. As usually, by $L_X(\theta)$ we denote the likelihood $\theta \mapsto p_\theta(X)$ whereas $l_X(\theta)$ stands for the log-likelihood $\ln(L_X(\theta))$. Finally, $\hat{\theta}_{ML}(X)$ denotes the **Maximum Likelihood Estimator** (MLE) of θ, i.e., $\hat{\theta}_{ML}(X) = \arg\ \max_{\theta\in\Theta} L_X(\theta)$. MLE is a technique which helps to determine the parameters of distribution that describe the given data the best.

- Let $\mathbb{P}(x|\theta)$ be a probability model, where x is a data item, θ is a vector of the model parameters, and ∇_θ the gradient operator with respect to θ. Then the Fisher score is defined as $U_x = \nabla_\theta \log_e \mathbb{P}(x|\theta)$.

  The Fisher score gives us an embedding into the feature space $\mathbb{R}^N$. The **Fisher kernel** refers to the inner product in this space, and is defined as $K(x_i, x_j) = U_{x_i}^T I^{-1} U_{x_j}$, where I is the Fisher information matrix.

- **Fisher information matrix** is defined as $I(\theta) = \mathbb{E}[\frac{d}{d\theta} l_X(\theta)(\frac{d}{d\theta} l_X(\theta))^T]$. It shows how much information about θ is in the observation.

- A function $f : 2^E \to \mathbb{R}_+$, where E is a finite set, is **submodular** if for all $S \subseteq T \subseteq E$, we have

$$f(T \cup \{l\}) - f(T) \leq f(S \cup \{l\}) - f(S)$$

  for all $l \in E \setminus T$.

- **Greedy** is an algorithm paradigm that builds up a solution piece by piece, always choosing the next piece that offers the most obvious and immediate benefit.

- **Cosine similarity** measures the similarity between two vectors of an inner product space. Given two vectors A and B, cosine similarity is defined as follows: $\cos\theta := \dfrac{A \cdot B}{||A||\,||B||}$.

- Let $f : \mathbb{R}^n \to \mathbb{R}$ be a scalar field. **Gradient** $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$ is a vector, such that $(\nabla f)_j = \partial f / \partial x_j$.

- **Hessian** is the matrix of second order mixed partials of a scalar field: $H_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j}$.

- Let $\mathbf{A}$ be a $m \times n$ matrix and $\mathbf{B}$ be a $p \times q$ matrix. **Kronecker product** $\mathbf{A} \otimes \mathbf{B}$ is defined as $pm \times qn$ block matrix:

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} a_{1,1}\mathbf{B} & \cdots & a_{1,n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m,1}\mathbf{B} & \cdots & a_{m,n}\mathbf{B} \end{pmatrix}$$

- **One-step Newton approximation** estimates the change in parameters:

$$\theta(1-w) - \theta^*(1) \approx \nabla\theta_{N(t)}(w) \stackrel{def}{=} (H_{\lambda,1}(1-w))^{-1} g_1(w),$$

where $H_{\lambda,1}(1-w) = \Big(\sum_{i=1}^{n}(1-w_i)\nabla^2_\theta l(x_i, y_i; \theta^*(1))\Big) + \lambda I$ is the regularized empirical Hessian at $\theta^*(1)$ but reweighted after removing the subset w and λ corresponds to regularization.

- **Weight Decay** is a regularization technique applied to the weights of a neural network. It is used to prevent overfitting.

- **Optimizers** are algorithms or methods used to change the attributes of the neural network. In this paper Adaptive Moment Estimation (Adam) and Stochastic Gradient Descent (SGD) optimizers were used.

Robust statistics

As described in Hampel et al. (2011) and Huber (1992), **robustness** is insensitivity to small deviations from the assumptions. Robust statistics is used when the model is fitted by the majority of the data but there is a fraction of observations - called outliers or outlying observations - that follow a different distribution than the assumed one.

There are three main principles in robust statistics:

1. A functional approach - parameter is described as a functional.
2. Neighbourhood of probability distributions - behaviour of functional is studied for other models than the assumed one.
3. Equivariance properties - it is checked if estimator follows some equivariance properties.

Empirical risk minimization

Empirical risk minimization (ERM) is a principle that considers minimizing the empirical loss on observed data.

Let us consider input space X and output space Y. Let $h : X \to Y$ (hypothesis) be a function that outputs an object $y \in Y$, given $x \in X$. Also, let $L(h(x), y)$ be the loss function under consideration.

According to Vapnik (1992), expected value of the loss, given hypothesis $h(x)$, is as follows:

$$R(h) = \mathbb{E}[L(h(x), y)] = \int L(h(x), y) \, \mathrm{d}P(x, y),$$

here $P(x, y) = P(y|x)P(x)$ is a joint distribution of a random variate $(x, y) \in X \times Y$ and it is unknown.

As $R(h)$ cannot be computed, the solution is to replace it with empirical risk functional:

$$R_e(h) = \frac{1}{n} \sum_{i=1}^{n} L(h(x_i), y_i).$$

Therefore, the ERM is described as optimization problem:

$$h^* = \arg\min_h R_e(h).$$

Implicit function theorem

Let F be a real-valued continuously differentiable function defined in a neighbourhood of $(X_0, Y_0) \in \mathbb{R}^2$. Suppose F satisfies the two conditions:

$$F(X_0, Y_0) = Z_0,$$
$$\frac{\partial F}{\partial Y}(X_0, Y_0) \neq 0.$$

Then there exist open intervals U and V, with $X_0 \in U$, $Y_0 \in V$, and a unique function $G : U \to V$ satisfying

$$F[X, G(X)] = Z_0,$$

for all $X \in U$, and this function F is continuously differentiable with

$$G'(X_0) = -\Big[\frac{\partial F}{\partial X}(X_0, Y_0)\Big] \Big/ \frac{\partial F}{\partial Y}(X_0, Y_0).$$

Activation functions

An **activation function** in a neural network defines how the weighted sum of the input is transformed into an output from a node or nodes in a layer of the network. It helps the neural network to use important information while suppressing irrelevant data points. Activation functions are differentiable as neural networks are typically trained using the backpropagation.

In this paper we use two activation functions: Sigmoid Linear Unit (SiLU) and Rectified linear unit (ReLU). They are compared in Figure 2.

SiLU is defined as:

$$f(x) = \frac{x}{1 + e^{-x}}$$

ReLu is defined as:

$$f(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$$

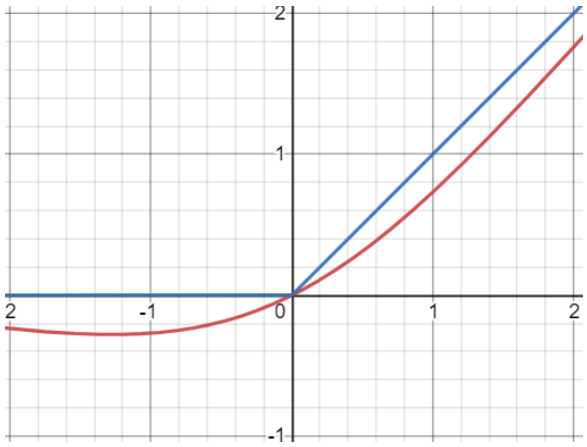

Figure 2: Activation functions: SiLU - red, ReLU - blue

Models

Models that are used in this paper:

- **Logistic regression** is a type of statistical analysis that models the probabilities for classification problems with two possible outcomes.
- **Convolutional Neural Network (CNN)** is a type of artificial neural network that requires a convolutional layer and can have other types of layers, such as nonlinear or pooling. In this paper we use three different CNNs: CNN for binary classification, shallow CNN, EfficientNet and ResNet-50.

Explainable AI (XAI)

Explainability of AI has not been clearly defined yet. It can be viewed as characteristics of a model that are used for demystifying model's internal functions. Explainability is related to interpretability: if operations of the systems are understood by human, then interpretable systems are called explainable. By Adadi and Berrada (2018) reasons of XAI are given:

- explain to justify,
- explain to control,
- explain to improve,
- explain to discover.

Even though interpretability concerns mathematical interpretation of how inputs are mapped to outputs, the explainability is considered to be under discussion in multiple disciplines, including social science and philosophy.

By Arrieta et al. (2020), XAI is divided into two classes: transparent models (e.g. logistic/linear regression, decision trees, Bayesian models) and post-hoc explainability. The later is divided into model-agnostic (e.g. feature relevance explanation) and model-specific (e.g. visual explanations of convolutional neural networks).

As concept of explainability is still on ongoing discussion, in this paper we compare different methods with each other and their results are defined as less/more intuitive based on their differences.

3. Literature review

Influence functions are a method from robust statistics. The main goal of IF is to estimate the effect of training points on a model's predictions by using first-order Taylor approximations. IF were first addressed in the 70s and 80s (Hampel (1974), Cook and Weisberg (1982)). Mostly, IF were applied in outliers detection (Campbell (1978)). In this paper, the criteria for examining multivariate data for outliers in discriminant analysis is developed using influence functions.

IF applications in explaining black box models was first introduced by Koh and Liang (2017). Some additional work followed (Koh et al. (2019)) where the effect of computations of corresponding groups were analysed. The setting of removing a constant fraction α of the training data has been empirically studied. Analysis centres on the one-step Newton approximation. Additional insights were proposed by Basu et al. (2020b). Because of the effect that removal of a large group of training points might have on model's parameters, the second-order approximations are studied in this paper.

Alternative methods for computing importance of training points without computing parameters use Data Shapley values (Ghorbani and Zou (2019)). In this proposed framework, the value depends on the three properties: learning algorithm, evaluation metric and other points in the training set. The Shapley values come from game theory, where game can be understood as collaborative game, the task as prediction, the gain as the distance between the prediction and baseline prediction, and the players are the features.

In Khanna et al. (2019), Fisher kernels were used in order to define sets of influential training points. The main goal there is to approximate the empirical test data distribution using samples from the training data. Therefore, all the points in the space induced by the Fisher kernels are embedded. If loss function in this method is negative log-likelihood, it recovers IF.

Additional insights about IF and their value in explainability of black box models were introduced. In Barshan et al. (2020), the idea of constraints for the change of parameters was proposed. Basu et al. (2020a) discussed the diverse influential points and suggested the alternative way of selecting regions without repetition.

In this paper, we will more deeply discuss some of the alternative methods mentioned above and how they work on models were IF does not return understandable examples from training set.

4. Influence functions

4.1 Background

Let us consider a ML model with parameters $\theta \in \Theta \subset \mathbb{R}^k$. Then let X be model's input space (e.g. images) and Y be the output space (e.g. labels). Moreover, let a set $S = (z_1, z_2, ..., z_n)$, where $z_i = (x_i, y_i) \in X \times Y$, be a training set. Then for any z and θ let $L(z, \theta)$ be a loss and $\frac{1}{n}\sum_{i=1}^{n} L(z_i, \theta)$ be the empirical risk.

The parameters are learnt by weighted empirical risk minimization. For that reason each training point z_i is assigned a weight ω_i. Primarily, parameter $\theta^* = \theta(1/n, ..., 1/n)$ is obtained by weighting all points equally, that is $\omega_i = \frac{1}{n}$ for all i. Then we modify i'th weight so that $\theta^*_{i,\epsilon_i} = \theta(1/n, ..., 1/n + \epsilon_i, ..., 1/n)$, where ϵ_i is deviation from ω_i, meaning that $\epsilon_i = \omega_i - 1/n$. Therefore, when $\epsilon_i = 1/n$, then $\omega_i = 0$ and that corresponds to dropping the point z_i from S. In other words, we upweight a training instance by an infinitesimally small step ϵ. After these modifications the model needs to be retrained and its loss needs to be calculated.

As retraining for different weight configurations is prohibitively slow and computationally expensive, the influence functions from robust statistics (Cook and Weisberg (1980) and Hampel (1974)) can be used for determining first-order approximation of change in θ^* around $\epsilon_i = 0$.

Then according to Koh and Liang (2017), under the assumptions that:

1. $\theta(\omega_1, ..., \omega_n)$ is a differentiable function of weights at $\omega_1 = ... = \omega_n = 1/n$, and
2. $L(z, \theta)$ is strictly convex and twice continuously differentiable in θ

the influence of z_i on test sample $z_{test} = (x_{test}, y_{test})$ is defined to be:

$$I_{test,i} \stackrel{def}{=} -\left.\frac{dL(z_{test}, \theta^*_{i,\epsilon_i})}{d\epsilon_i}\right|_{\epsilon_i=0} = -g_{test}^T \left.\frac{d\theta^*_{i,\epsilon_i}}{d\epsilon_i}\right|_{\epsilon_i=0}, \tag{1}$$

where $g_{test} = \nabla_{\theta^*} L(z_{test}, \theta^*)$ is the loss gradient with respect to the parameters for the upweighted training instance.

4.2 Influence for General Loss Functions

By applying implicit function theorem (Cook and Weisberg (1982)) it can be shown that:

$$\left.\frac{d\theta^*_{i,\epsilon_i}}{d\epsilon_i}\right|_{\epsilon_i=0} = -H_{\theta^*}^{-1} g_i, \tag{2}$$

where $g_i = \nabla_{\theta^*} L(z_i, \theta^*)$ and $H_{\theta^*} = \frac{1}{n}\sum_{i=1}^{n} \nabla^2_{\theta^*} L(z_{test}, \theta^*)$ is a Hessian matrix - a second derivative of the loss with respect to the model's parameters, meaning that Hessian matrix describes the curvature of the loss.

Then equation 1 can be rewritten in this way:

$$I_{test,i} = g_{test}^T H_{\theta^*}^{-1} g_i. \tag{3}$$

The above formula demonstrates how the influence depends on the gradients for the training and test loss.

Then by applying first-order Taylor series approximation

$$f(x) = \sum_{n=0}^{\infty} \frac{f^n(a)}{n!}(x-a)^n$$

and noticing that $\theta^*_{i,\epsilon_i}\big|_{\epsilon_i=0} = \theta^*$, the change of parameters' weights can be written as:

$$\theta^*_{i,\epsilon_i} - \theta^* \approx \theta^* - \theta^* + \left.\frac{d\theta^*_{i,\epsilon_i}}{d\epsilon_i}\right|_{\epsilon_i=0} \epsilon_i = -H^{-1}_{\theta^*} g_i \epsilon_i. \tag{4}$$

Similarly, the difference in loss is approximated like this:

$$L(z_{test}, \theta^*_{i,\epsilon_i}) - L(z_{test}, \theta^*) \approx \left.\frac{dL(z_{test}, \theta^*_{i,\epsilon_i})}{d\epsilon_i}\right|_{\epsilon_i=0} \epsilon_i = -g^T_{test} H^{-1}_{\theta^*} g_i \epsilon_i. \tag{5}$$

5. Why don't influence functions work all the time?

In order to define influence of test sample there were two strict assumptions applied in Section 4. In this section we will discuss how assumption of convexity cause problems while applying influence functions on some of the ML models. Additionally, tendency to return outliers as the most influential is also analysed.

5.1 Convexity

An importance of convexity in loss function comes from a convex function definition. As it has an increasing first derivative, making it appear to bend upwards (all pairs of points on the function graph lies above function curve), the minimum of a convex function is also a global minimum.

This is a usual case in deep learning that loss function violates convexity assumption. However, expressing learning models in a way that non-convex optimization problems are solved, gives tremendous modelling power (Jain and Kar (2017)).

The loss function in deep learning models might have a number of loss maxima and minima. Example of such case is demonstrated in Figure 3). It happens because of the permutation of the nodes which are assigned to different parameters to minimize loss function. Therefore, it is possible to have multiple solutions of the parameters resulting to the same loss in the subsequent layers. Nevertheless, having multiple minima in the search space is not a problem in deep learning as all of them happen to be very close.

Although, having non-convex loss function, is not proven to do a lot damage for evaluation of model's parameters, the problem for influence function still occurs, as it was described in Basu et al. (2020a). In that paper, it has been empirically

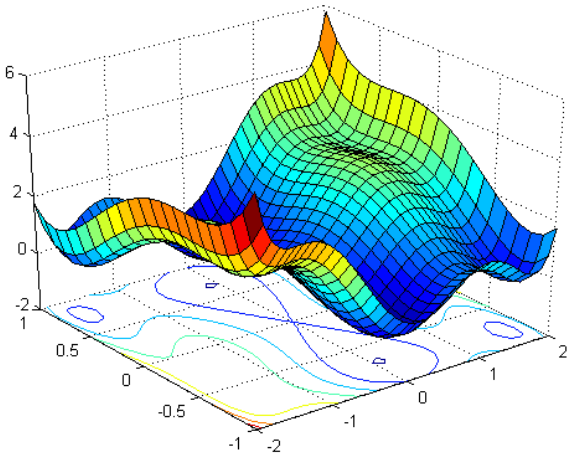

Figure 3: Example of function with multiple minima

demonstrated that network architecture, its depth and width, related to non-convex loss functions, cause the inaccuracy of explainability using IF.

5.2 Outliers and mislabelled points

Influence functions initially have been used as an empirical diagnostic tool in order to identify outliers (Cook and Weisberg (1982), Campbell (1978)). Therefore, IF applications as a solution for ML models explainability can be misleading for end users as returned points cannot be understood intuitively.

According to Barshan et al. (2020), using IF, training points that are defined as the most influential are also highly overlapping. This means that small set of training points that greatly affects the predictions is the same for many different inputs. These points are said to have global effect.

6. Alternatives

In this section we will briefly discuss the methods suggested as the alternatives to influence function and solving the problems mentioned in the previous section.

6.1 RelatIF

In Barshan et al. (2020), an alternative representation of IF in terms of Fisher information using cross entropy loss and maximum likelihood estimation was presented.

The main goal of Relative influence functions (RelatIF) is to constrain the change of model parameters. This means that local influence that a testing point has on a prediction is compared to its global effect on the model.

6.1.1 Influence of Maximum likelihood

Firstly, the influence in terms of Fisher information is defined. Let us consider a parametric statistical model $p_\theta(y|x)$. Then let $l((x,y),\theta) = -\log p_\theta(y|x)$.

According to Ting and Brochu (2018), it can be shown that influence function then can be written in this form:

$$I_{test,i} = g_{test}^T F_{\theta^*}^{-1} g_i, \tag{6}$$

where $F_\theta = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{p_\theta} \nabla_\theta \log p_\theta(y|x_i) \nabla_\theta \log p_\theta(y|x_i)^T$ is the Fisher information matrix.

Then after applying Taylor series we get:

$$\theta_{i,\epsilon_i} - \theta^* \approx -F_{\theta^*}^{-1} g_i \epsilon_i, \tag{7}$$

and

$$l(z_{test}, \theta^*_{i,\epsilon_i}) - l(z_{test}, \theta^*) \approx -g_{test}^T F_{\theta^*}^{-1} g_i \epsilon_i. \tag{8}$$

6.1.2 Constraints

In order to get top-k influential examples $|I_{test,i}|$ is maximized. In Barshan et al. (2020), there was introduced alternative method with additional constraints on how the model may change. Now we briefly introduce these constraints.

θ-Relative Influence

Firstly, the direct constraint of model parameters is presented. Therefore, the below problem is being solved:

$$\begin{aligned} &\arg\max_{i\in\{1,\dots,n\}} \max_{\epsilon_i} |l(z_{test}, \theta^*_{i,\epsilon_i}) - l(z_{test}, \theta^*)| \\ &s.t. ||\theta_{i,\epsilon_i^*} - \theta||^2 \leq \delta^2. \end{aligned} \tag{9}$$

In Barshan et al. (2020) it is shown that equation 9 can be rewritten as follows:

$$\arg\max_{i\in\{1,\dots,n\}} \frac{|I_{test,i}|}{||H_{\theta^*}^{-1} g_i||} \tag{10}$$

and the answer, the θ-relative influence of training example z_i on the loss of test sample z_{test} is defined as:

$$\frac{I_{test,i}}{||H_{\theta^*}^{-1} g_i||}.$$

l-Relative Influence

In this part constraint for model's interpretation in terms of maximum likelihood with the output $p_\theta(y|x)$ is analysed. The change in squared loss is of interest.

Similarly as before, Barshan et al. (2020) describes the problem:

$$\begin{aligned} &\arg\max_{i\in\{1,\dots,n\}} \max_{\epsilon_i} |l(z_{test}, \theta^*_{i,\epsilon_i}) - l(z_{test}, \theta^*)| \\ &s.t. \mathbb{E}_{p_{\theta^*}}(l(z, \theta^*_{i,\epsilon_i}) - l(z, \theta^*))^2 \leq \delta^2. \end{aligned} \tag{11}$$

The problem is rewritten:

$$\arg\max_{i\in\{1,\dots,n\}} \frac{|I_{test,i}|}{\sqrt{I_{i,i}}} \tag{12}$$

and the answer, the l-relative influence of training example z_i on the loss of test sample z_{test} is defined as:

$$\frac{I_{test,i}}{\sqrt{I_{i,i}}}.$$

Interpretation

By rewriting influence like this:

$$I_{test,i} = \big\langle -g_{test}, \frac{d\theta^*_{i,\epsilon_i}}{d\epsilon_i} \big\rangle = \big\langle -g_{test}, -H_{\theta^*}^{-1} g_i \big\rangle,$$

where $\langle \cdot, \cdot \rangle$ is the inner product operator, one sees that $|I_{test,i}|$ is equal to the projection length of the change in parameters vector onto the test sample's loss gradient. θ-RelatIF, however, uses cosine similarity projection. Similarly, l-RelatIF uses cosine similarity under the inner product $\langle a, b \rangle = a^T F_{\theta^*}^{-1} b$.

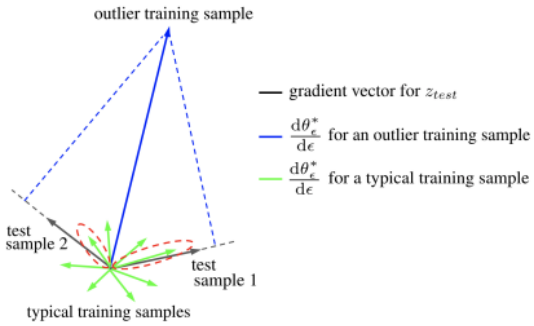

Figure 4: Geometric interpretation of RelatIF. Figure from Barshan et al. (2020)

In Figure 4 thick blue line corresponds to the outlier training example that is the most influential. Meanwhile, RelatIF returns red dotted circles. This happens because RelatIF uses cosine similarity in order to identify influential points.

6.2 DIVINE

In Bhatt et al. (2021), alternative method is suggested – DIVINE (DIVerseINfluEntial) training points are selected. Differently from RelatIF, DIVINE does not solve a problem of atypical data, it focuses on receiving influential data points from different regions, so that the same region would not be selected repeatedly.

The main objective described in Bhatt et al. (2021) is as follows:

$$\max_{S \in D, |S|=m} I(S) + \gamma R(S). \tag{13}$$

Here:

- m are influential points,
- D is a given data set,
- S is a subset of m points from data set D,
- $I(S)$ is a normalized function, $(I(\emptyset) = 0)$,
- γ controls the trade-off between two terms. Therefore, when $\gamma = 0$ we get traditional m points that are returned by IF.

The solution for this optimization problem is m diverse influential (DIVINE) points. In Bhatt et al. (2021), there are three submodular $R(S)$ expressions suggested:

1. Sum-Redundancy $R_{SR}(S) = \kappa - \sum_{u,v \in S} \phi(u, v)$, where $\kappa = \sum_{u,v \in D} \phi(u, v)$, is penalty-based diversity. It finds diverse m influential points and penalizes similarity between points in S;

2. Facility-Location $R_{FL}(S) = \sum_{u \in D} \max_{v \in S} \phi(u, v)$ maximizes the average similarity between a training point and its most similar point in S;

3. Maximum mean discrepancy $R_{MMD}(S) = c_1 \sum_{u \in D, v \in S} \phi(u, v) - c_2 \sum_{u, v \in S} \phi(u, v)$, where $c_1 = \frac{2}{n|S|}$ and $c_2 = \frac{1}{|S|^2}$, ensures the m points are similar to the training data while being different from each other.

Here ϕ is the radial basis function kernel and $I(S) = \sum_{i \in S} I_i$ is the sum of important scores of points in S.

For solving optimization problem, the greedy algorithm is given in Bhatt et al. (2021):

1. Influence I_i for x_i is calculated for all $x_i \in D$.

2. A set of m DIVINE points is found by computing $\arg\max_{x_i \in D \setminus S} [I_S + I_i + \gamma R(S)]$.

3. I_S is set to be the sum of all I_i.

7. Experiments

In order to identify differences between estimations of influential points, few models are compared: logistic regression trained on MNIST data set (LeCun and Cortes (2010)), CNN for binary classification, and three CNN models trained on CIFAR-10 data set (Krizhevsky et al.). We show how deeper architecture makes an impact on explainability. Moreover, we conduct the experiments using different loss functions and varying weight-decay regularization. While training models, we seed all sources of randomness to be able to compare results between methods.

For both methods we select top 3 influential points in all cases. In the last subsection we present the running times. In all cases, to compute the most influential points for one class took less than 24 hours.

As it was discussed in Barshan et al. (2020), for illustrative examples we use θ-RelatIF as it produces similar results as RelatIF.

7.1 Logistic regression

To find influential points in models where exact Hessian can be computed, first setup included logistic regression trained on MNIST data set. Model is trained with stochastic gradient decent, without weight-decay. In Figure 5 we can see top 3 influential points for two randomly selected test points: 3 and 8.

Then we select 1000 test points and their most influential training points. In Table 1 the loss of retraining the model (i.e., leave-one-out retraining) is compared. In the results, it is demonstrated that change in loss after removing IF points is a little higher than removing RelatIF points.

Loss of retrained models	
Model	Loss
Originl dataset	1.567
Dataset without IF	1.599
Dataset without RelatIF	1.577

Table 1: Loss of retrained models without top 1000 randomly selected influential points

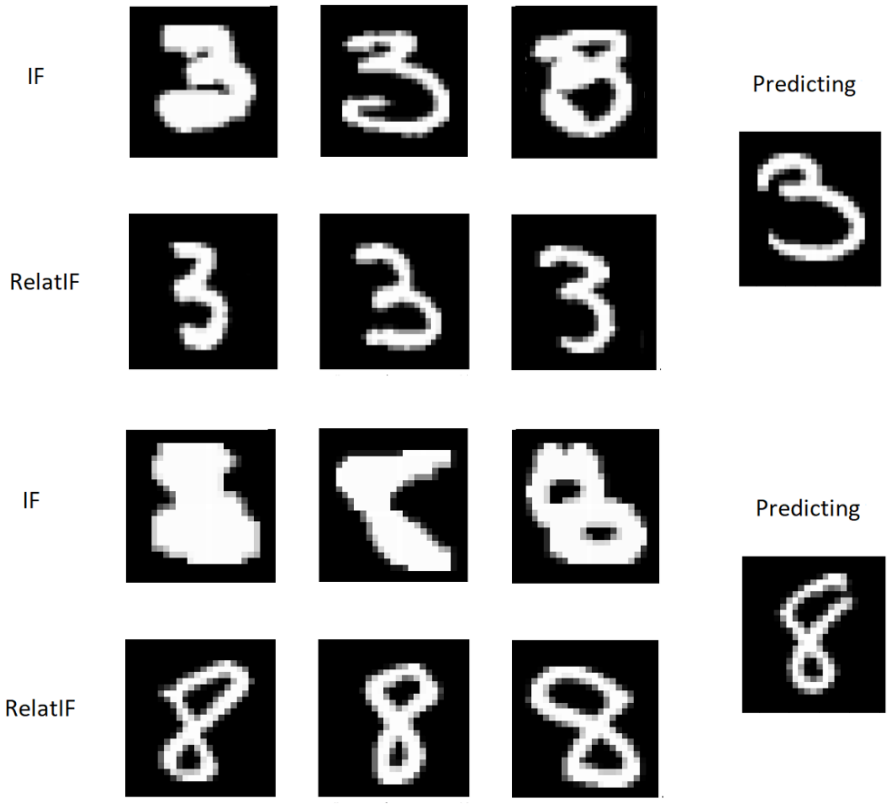

Figure 5: Top 3 most influential points produced by IF and RelatIF

7.2 CNN for binary classification

In this subsection we examine the performance of IF and RelatIF using CNN architecture for binary classification. As data set we used two classes of images of motorcycles and apples. In Figures 6 and 7 we can see the comparison of influential points produced by IF and RelatIF using binary cross entropy and negative log-likelihood respectively. Experiments with hinge loss have not returned significant results as influence was equal to 0 for most of the data points.

We also compare the most influential points without weight-decay regularization, meaning we are interested in effect of adding a small penalty to the loss function. Similar results for explainability were returned (Figure 8).

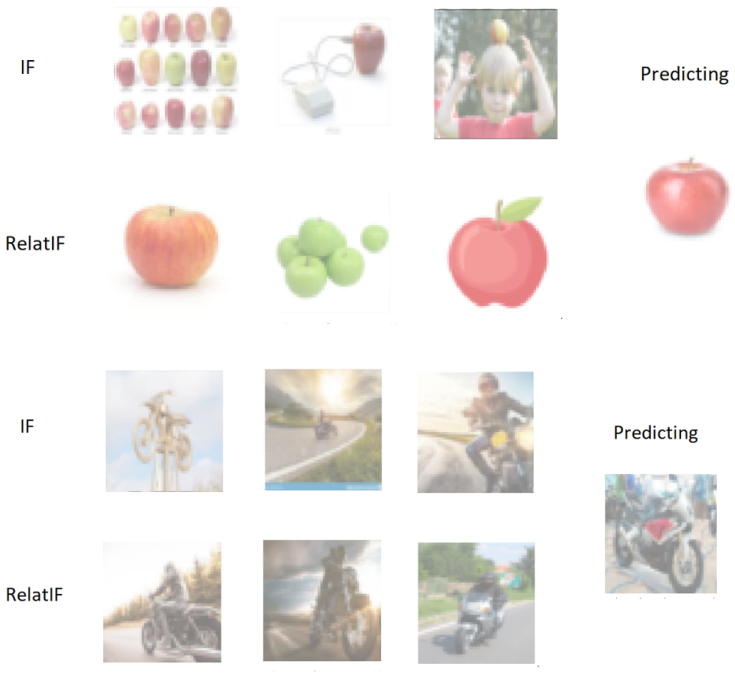

Figure 6: Top 3 most influential points produced by IF and RelatIF using BCE loss

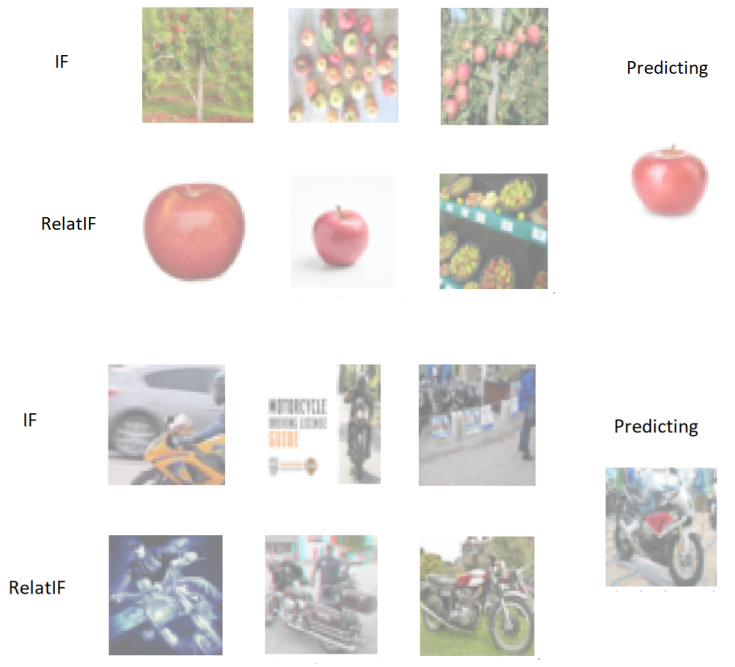

Figure 7: Top 3 most influential points produced by IF and RelatIF using NLL loss

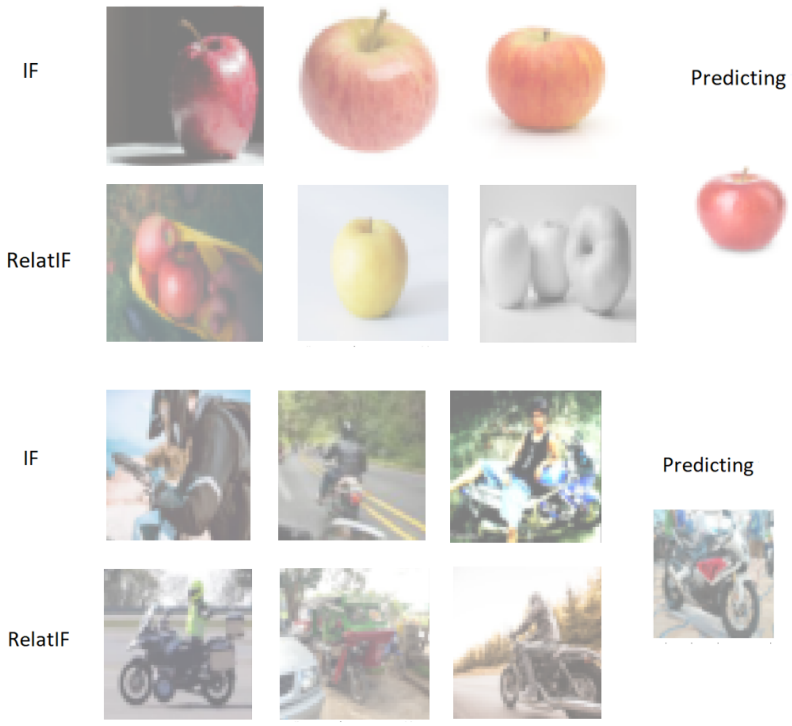

Figure 8: Top 3 most influential points produced by IF and RelatIF using BCE loss with weight decay

7.3 CNN for classification

In this setup we analyse three CNN models trained on CIFAR-10 data set: shallow CNN, EfficientNet and ResNet-50. We can see top 3 most influential points in each case.

EfficientNet is a convolutional neural network architecture and scaling method that uniformly scales all dimensions of depth/width/resolution using a compound coefficient. ResNet-50 model is a deep residual convolutional neural network that is 50 layers deep.

Models were selected to check the effect that network's architecture has in determining influential points. Networks vary in depth, weight, activation functions (ReLu and SiLu) and optimizers (Adam and SGD). Same Cross Entropy loss was used for all models.

It can be observed that in shallow CNN RelatIF returned influential points with higher quality. However, with EfficientNet and ResNet-50 results do not return training points that would be easy to understand for end users.

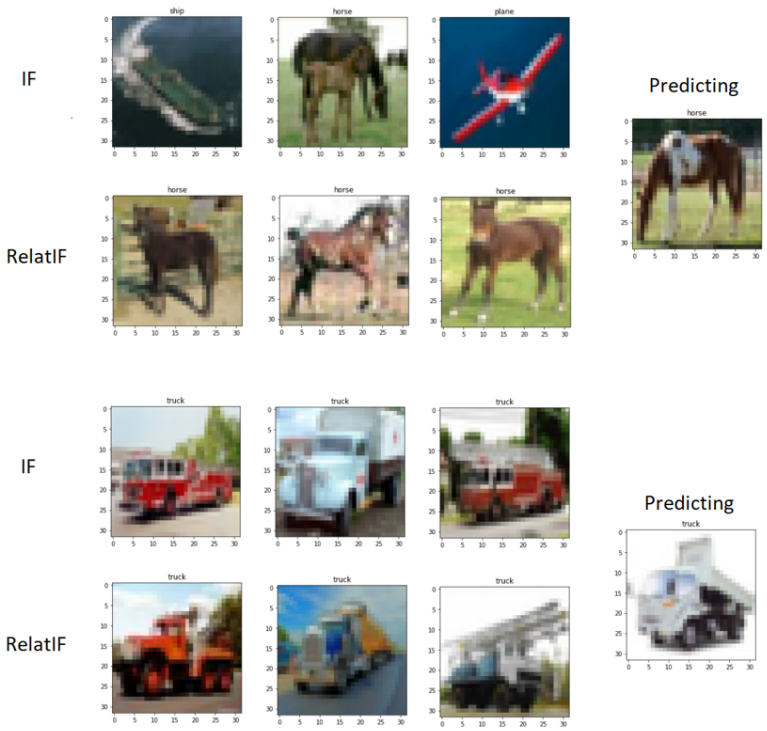

Figure 9: Small CNN results for influential points

Figure 10: EfficientNet results for influential points

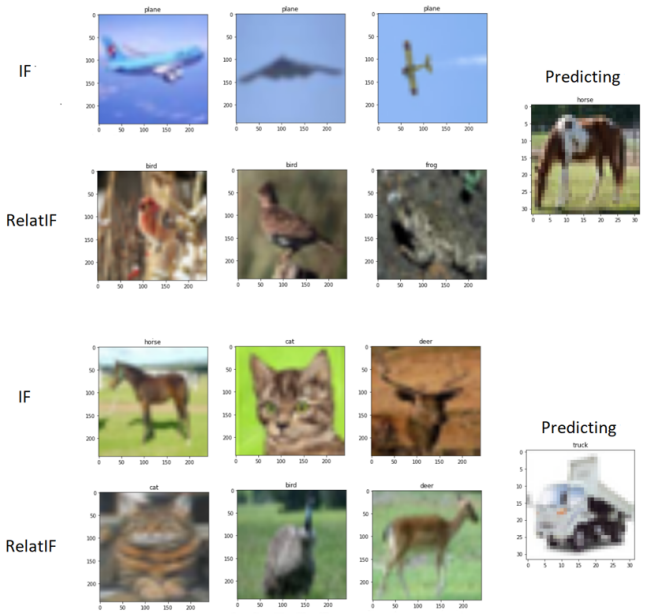

Figure 11: ResNet-50 results for influential points

7.4 Running times

In Table 2 the times for calculating influence function and relative influence functions on different models are compared.

Average running times			
Model	IF	RelatIF	Training sample size
Logistic Regression	168.917	750.457	50,000
CNN for binary classification	17.342	46.841	612
CNN	424.087	11,731.303	10000
EfficientNet	139.767	10759.267	3,500
ResNet-50	835.111	34960.431	50,000

Table 2: Average running time (in seconds) for calculating influential points of one test sample.

It can be noticed that RelatIF requires additional approximation in order to reduce the time for computations. As suggested by Barshan et al. (2020), Kronecker-factored Approximate Curvature (K-FAC) could be considered (Martens and Grosse (2015)). It is based on approximation of Fisher information matrix of neural network. K-FAC can be derived by approximating various large blocks of the Fisher as being the Kronecker product of two much smaller matrices.

8. Conclusions

In this paper, we tried to answer a question why the model makes a certain prediction for a particular test sample based on the training data set. We compared relative influence functions (RelatIF) in models where IF were found to encounter difficulties. In order to demonstrate this, we compared different models (logistic regression, convolutional neural network for binary classification, small CNN, EfficientNet and ResNet-50 for classification), using multiple loss functions. It was found that influential points can be intuitively understood by end users in models were exact Hessian can be computed with both, IF and RelatIF. However, influential points do not appear to explain model when deeper neural network models are used. In binary classification case, we observe that regularizer (weight-decay) used in model has made no significant change for intuitive explanations.

In addition to that, we presented the running times for RelatIF and IF. RelatIF were found to run at least several times longer than IF in small models and 20 to 40 times longer than IF in large models. In order to reduce the time, approximations motivated by K-FAC could be considered in the future analysis. However, additional approximations might impact the quality of estimates.

The experiments were not conducted on the third, DIVINE (DIVerseINfluEntial), method as it was originally planned due to technical problems. Mastering of unknown software and debugging the code took more time than expected at first. Additionally, calculations of inverse Hessian-Vector product that are used in influence functions estimation, are computationally expensive.

References

Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160, 2018.

Tanveer Ahmad and Huanxin Chen. A review on machine learning forecasting growth trends and their real-time applications in different energy systems. *Sustainable Cities and Society*, 54:102010, 2020.

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115, 2020.

Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. Relatif: Identifying explanatory training samples via relative influence. In *International Conference on Artificial Intelligence and Statistics*, pages 1899–1909. PMLR, 2020.

Samyadeep Basu, Philip Pope, and Soheil Feizi. Influence functions in deep learning are fragile. *arXiv preprint arXiv:2006.14651*, 2020a.

Samyadeep Basu, Xuchen You, and Soheil Feizi. On second-order group influence functions for black-box predictions. In *International Conference on Machine Learning*, pages 715–724. PMLR, 2020b.

Umang Bhatt, Alice Xiang, Shubham Sharma, Adrian Weller, Ankur Taly, Yunhan Jia, Joydeep Ghosh, Ruchir Puri, José MF Moura, and Peter Eckersley. Explainable machine learning in deployment. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 648–657, 2020.

Umang Bhatt, Isabel Chien, Muhammad Bilal Zafar, and Adrian Weller. Divine: Diverse influential training points for data visualization and model refinement. *arXiv preprint arXiv:2107.05978*, 2021.

Adrien Bibal, Michael Lognoul, Alexandre De Streel, and Benoît Frénay. Legal requirements on explainability in machine learning. *Artificial Intelligence and Law*, 29(2):149–169, 2021.

Norm A Campbell. The influence function as an aid in outlier detection in discriminant analysis. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 27(3):251–258, 1978.

R Dennis Cook and Sanford Weisberg. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 22(4):495–508, 1980.

R Dennis Cook and Sanford Weisberg. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, pages 2242–2251. PMLR, 2019.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

Frank R Hampel. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393, 1974.

Frank R Hampel, Elvezio M Ronchetti, Peter J Rousseeuw, and Werner A Stahel. *Robust statistics: the approach based on influence functions*, volume 196. John Wiley & Sons, 2011.

Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992.

Prateek Jain and Purushottam Kar. Non-convex optimization for machine learning. *arXiv preprint arXiv:1712.07897*, 2017.

Rajiv Khanna, Been Kim, Joydeep Ghosh, and Sanmi Koyejo. Interpreting black box predictions using fisher kernels. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3382–3390. PMLR, 2019.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894. PMLR, 2017.

Pang Wei Koh, Kai-Siang Ang, Hubert HK Teo, and Percy Liang. On the accuracy of influence functions for measuring group effects. *arXiv preprint arXiv:1905.13289*, 2019.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL `http://yann.lecun.com/exdb/mnist/`.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417. PMLR, 2015.

Daniel Ting and Eric Brochu. Optimal subsampling with influence functions. In *Advances in neural information processing systems*, pages 3650–3659, 2018.

Vladimir Vapnik. Principles of risk minimization for learning theory. In *Advances in neural information processing systems*, pages 831–838, 1992.

Xizhao Wang, Yanxia Zhao, and Farhad Pourpanah. Recent advances in deep learning, 2020.