On Second-Order Group Influence Functions for Black-Box Predictions

Samyadeep Basu [1] Xuchen You [1] Soheil Feizi [1]

Abstract

With the rapid adoption of machine learning systems in sensitive applications, there is an increasing need to make black-box models explainable. Often we want to identify an *influential* group of training samples in a particular test prediction for a given machine learning model. Existing influence functions tackle this problem by using first-order approximations of the effect of removing a sample from the training set on model parameters. To compute the influence of a group of training samples (rather than an individual point) in model predictions, the change in optimal model parameters after removing that group from the training set can be large. Thus, in such cases, the first-order approximation can be loose. In this paper, we address this issue and propose second-order influence functions for identifying influential groups in test-time predictions. For linear models, across different sizes and types of groups, we show that using the proposed second-order influence function improves the correlation between the computed influence values and the ground truth ones. We also show that second-order influence functions could be used with optimization techniques to improve the selection of the most influential group for a test-sample.

1. Introduction

Recently, there has been a rapid and significant success in applying machine learning methods to a wide range of applications including vision (Szeliski, 2010), natural language processing (Sebastiani, 2002), medicine (Lundervold and Lundervold, 2018), finance (Lin, Hu, and Tsai, 2012), etc. In sensitive applications such as medicine, we would like to explain test-time model predictions to humans. An important question is : *why the model makes a certain prediction for a particular test sample.* One way to address this is to trace back model predictions to its training data. More specifically, one can ask which training samples were the most influential ones for a given test prediction.

Influence functions (Cook and Weisberg, 1980) from robust statistics measure the dependency of optimal model parameters on training samples. Previously (Koh and Liang, 2017) used first-order approximations of influence functions to estimate how much model parameters would change if a training point was up-weighted by an infinitesimal amount. Such an approximation can be used to identify most influential training samples in a test prediction. Moreover, this approximation is similar to the leave-one-out re-training, thus the first-order influence function proposed in (Koh and Liang, 2017) bypasses the expensive process of repeated re-training the model to find influential training samples in a test-time prediction.

In some applications, one may want to understand how model parameters would change when large groups of training samples are removed from the training set. This could be useful to identify groups of training data which drive the decision for a particular test prediction. As shown in (Koh, Ang, Teo, and Liang, 2019a), finding influential groups can be useful in real-world applications such as diagnosing batch effects (Yang, Li, Qian, Wilhelmsen, Shen, and Li, 2019), apportioning credit between different data sources (Arrieta-Ibarra, Goff, Jiménez-Hernández, Lanier, and Weyl, 2018), understanding effects of different demographic groups (Chen, Johansson, and Sontag, 2018) or in a multi-party learning setting (Hayes and Ohrimenko, 2019). (Koh et al., 2019a) approximates the group influence by sum of first-order individual influences over training samples in the considered group. However, removal of a large group from training can lead to a large perturbation to model parameters. Therefore, influence functions based on first-order approximations may not be accurate in this setup. Moreover, approximating the group influence by adding individual sample influences ignores possible cross correlations that may exist among samples in the group.

In this paper, we relax the first-order approximations of current influence functions and study how second-order approximations can be used to capture model changes when a potentially large group of training samples is up-weighted. Considering a training set $\mathcal{S}$ and a group $\mathcal{U} \subset \mathcal{S}$, existing

[1] Department of Computer Science, University of Maryland-College Park. Correspondence to: Samyadeep Basu <sbasu12@cs.umd.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020. Copyright 2020 by the author(s).

first-order approximations of the group influence function (Koh et al., 2019a) can be written as the sum of first-order influences of individual points. That is,

$$\mathcal{I}^{(1)}(\mathcal{U}) = \sum_{i=1}^{|\mathcal{U}|} \mathcal{I}_i^{(1)}$$

where $\mathcal{I}^{(1)}(\mathcal{U})$ is the first-order group influence function and $\mathcal{I}_i^{(1)}$ is the first-order influence for the i^{th} sample in $\mathcal{U}$. On the other hand, our proposed second-order group influence function has the following form:

$$\mathcal{I}^{(2)}(\mathcal{U}) = \mathcal{I}^{(1)}(\mathcal{U}) + \mathcal{I}^{'}(\mathcal{U})$$

where $\mathcal{I}^{'}(\mathcal{U})$ captures informative cross-dependencies among samples in the group and is a function of gradient vectors and the Hessian matrix evaluated at the optimal model parameters. We present a more precise statement of this result in Theorem 1. We note that the proposed second-order influence function can be computed efficiently even for large models. We discuss its computational complexity in Section 6.

Our analysis shows that the proposed second-order influence function captures model changes efficiently even when the size of the groups are relatively large or the changes to the model parameters are significant as in the case of groups with similar properties. For example, in an MNIST classification problem using logistic regression, when 50% of the training samples are removed, the correlation between the ground truth estimate and second-order influence values improves by over 55% when compared to the existing first-order influence values. We note that higher-order influence functions have been used in statistics (James, Lingling, Eric, and van der Vaart, 2017) for point and interval estimates of non-linear functionals in parameteric, semi-parametric and non-parametric models. However, to the best of our knowledge, this is the first time, higher-order influence functions are used for the interpretability task in the machine learning community.

Similar to (Koh and Liang, 2017) and (Koh et al., 2019a), our main results for the second-order influence functions hold for linear prediction models where the underlying optimization is convex. However, we also additionally explore effectiveness of both first-order and second-order group influence functions in the case of deep neural networks. We observe that none of the methods provide good estimates of the ground-truth influence across different groups [1]. In summary, we make the following contributions:

- We propose second-order group influence functions that consider cross dependencies among the samples in the considered group.
- Through several experiments over linear models, across different sizes and types of groups, we show that the second-order influence estimates have higher correlations with the ground truth when compared to the first-order ones, especially when the changes to the underlying model is relatively large.
- We also show that our proposed second-order group influence function can be used to improve the selection of the most influential training group.

2. Related Works

Influence functions, a classical technique from robust statistics introduced by (Cook and Weisberg, 1980; Cook and Sanford, 1982) were first used in the machine learning community for interpretability by (Koh and Liang, 2017) to approximate the effect of upweighting a training point on the model parameters and test-loss for a particular test sample. In the past few years, there has been an increase in the applications of influence functions for a variety of machine learning tasks. (Schulam and Saria, 2019) used influence functions to produce confidence intervals for a prediction and to audit the reliability of predictions. (Wang, Ustun, and Calmon, 2019) used influence functions to approximate the gradient in order to recover a counterfactual distribution and increase model fairness, while (Brunet, Alkalay-Houlihan, Anderson, and Zemel, 2018) used influence functions to understand the origins of bias in word-embeddings. (Koh, Steinhardt, and Liang, 2019b) crafted stronger data poisoning attacks using influence functions. Influence functions can also be used to detect extrapolation (Madras, Atwood, and D'Amour, 2019) in certain specific cases, validate causal inference models (Alaa and Van Der Schaar, 2019) and identify influential pre-training points (Chen, Si, Li, Chelba, Kumar, Boning, and Hsieh, 2020). Infinitesimal jackknife or the delta method are ideas closely related to influence functions for linear approximations of leave-one-out cross validation (Jaeckel, 1972; Efron, 1992). Recently a higher-order instance (Giordano, Jordan, and Broderick, 2019) of infinitesimal jackknife (Jaeckel, 1972) was used to approximate cross-validation procedures. While their setting corresponding to approximations of leave-k-out re-training is relatively similar to our paper, our higher-order terms preserve the empirical weight distribution of the training data in the ERM and are derived from influence functions, while in (Giordano et al., 2019) instances of infinitesimal jackknife is used. These differences lead to our higher-order terms being marginally different than the one proposed in (Giordano et al., 2019). Our proposed second-order approximation for group influence function is additionally backed by a thorough empirical study across different settings in the case of linear models which has not yet been explored in prior works.

[1] Note that experiments of (Koh and Liang, 2017) focus only on the most influential individual training samples.

3. Background

We consider the classical supervised learning problem setup, where the task is to learn a function h (also called the hypothesis) mapping from the input space $\mathcal{X}$ to an output space $\mathcal{Y}$. We denote the input-output pair as $\{x, y\}$. We assume that our learning algorithm is given training examples $\mathcal{S} := \{z_i = (x_i, y_i)\}_{i=1}^m$ drawn i.i.d from some unknown distribution $\mathcal{P}$. Let Θ be the space of the parameters of considered hypothesis class. The goal is to select model parameters θ to minimize the empirical risk as follows:

$$\min_{\theta \in \Theta} \; L_{\emptyset}(\theta) := \frac{1}{|\mathcal{S}|} \sum_{z \in \mathcal{S}} \ell(h_\theta(z)), \tag{1}$$

where $|\mathcal{S}| = m$, denotes the cardinality of the training set, the subscript $\emptyset$ indicates that the whole set $\mathcal{S}$ is used in training and ℓ is the associated loss function. We refer to the optimal parameters computed by the above optimization as θ^*.

Let $\nabla_\theta L_\emptyset(\theta)$ and $H_{\theta^*} = \nabla_\theta^2 L_\emptyset(\theta)$ be the gradient and the Hessian of the loss function, respectively.

First, we discuss the case where we want to compute the effect of an *individual* training sample z on optimal model parameters as well as the test predictions made by the model. The effect or influence of a training sample on the model parameters could characterized by removing that particular training sample and retraining the model again as follows:

$$\theta^*_{\{z\}} = \arg\min_{\theta \in \Theta} \; L_{\{z\}}(\theta) = \frac{1}{|\mathcal{S}| - 1} \sum_{z_i \neq z} \ell(h_\theta(z_i)) \tag{2}$$

Then, we can compute the change in model parameters as $\Delta\theta = \theta^*_{\{z\}} - \theta^*$, due to removal of a training point z. However, re-training the model for every such training sample is expensive when $|\mathcal{S}|$ is large. Influence functions based on first-order approximations introduced by (Cook and Weisberg, 1980; Cook and Sanford, 1982) was used by (Koh and Liang, 2017) to approximate this change. Up-weighting a training point z by an infinitesimal amount ϵ leads to a new optimal model parameters, $\theta^\epsilon_{\{z\}}$, obtained by solving the following optimization problem:

$$\theta^\epsilon_{\{z\}} = \arg\min_{\theta \in \Theta} \; \frac{1}{|\mathcal{S}|} \sum_{z \in \mathcal{S}} \ell(h_\theta(z_i)) + \epsilon \ell(h_\theta(z)) \tag{3}$$

Removing a point z is similar to up-weighting its corresponding weight by $\epsilon = -\frac{1}{|\mathcal{S}|}$. The main idea used by (Koh and Liang, 2017) is to approximate $\theta^*_{\{z\}}$ by minimizing the first-order Taylor series approximation around θ^*. Following the classical result by (Cook and Weisberg, 1980), the change in the model parameters θ^* on up-weighting z can be approximated by the influence function (Koh and Liang, 2017) denoted by $\mathcal{I}$:

$$\mathcal{I}(z) = \frac{d\theta^\epsilon_{\{z\}}}{d\epsilon}|_{\epsilon=0} = -H_{\theta^*}^{-1} \nabla_\theta \ell\left(h_{\theta^*}(z)\right) \tag{4}$$

A detailed proof can be found in (Koh and Liang, 2017). Using the given formulation, we can track the change with respect to any function of θ^*. The change in the test loss for a particular test point z_t when a training point z is up-weighted can be approximated as a closed form expression:

$$\mathcal{I}(z, z_t) = -\nabla_\theta \ell(h_{\theta^*}(z_t))^T H_{\theta^*}^{-1} \nabla_\theta \ell(h_{\theta^*}(z)) \tag{5}$$

This result is based on the assumption (Koh and Liang, 2017) that the loss function $L(\theta)$ is strictly convex in the model parameters θ and the Hessian H_{θ^*} is therefore positive-definite. This approximation is very similar to forming a quadratic approximation around the optimal parameters θ^* and taking a single Newton step. However explicitly computing H_{θ^*} and it's inverse $H_{\theta^*}^{-1}$ is not required. Using the Hessian-vector product rule (Pearlmutter, 1994) influence functions can be computed efficiently.

4. Group Influence Function

Our goal in this section is to understand how the model parameters would change if a particular group of samples was up-weighted from the training set. However, up-weighting a group can lead to large perturbations to the training data distribution and therefore model parameters, which does not follow the small perturbation assumption of the first-order influence functions. In this section, we extend influence functions using second-order approximations to better capture changes in model parameters due to up-weighting a group of training samples. In Section 5, we show that our proposed second-order group influence function can be used in conjunction with optimization techniques to select the most influential training groups in a test prediction.

The empirical risk minimization (ERM) when we remove $\mathcal{U}$ samples from training can be written as:

$$L_{\mathcal{U}}(\theta) = \frac{1}{|\mathcal{S}| - |\mathcal{U}|} \sum_{z \in \mathcal{S} \setminus \mathcal{U}} \ell(h_\theta(z)) \tag{6}$$

To approximate how optimal solution of this optimization is related to θ^*, we study the effect of *up-weighting* a group of training samples on model parameters. Note that in this case, updated weights should still be a valid distribution, i.e. if a group of training samples has been up-weighted, the rest of samples should be down-weighted to preserve the sum to one constraint of weights in the ERM formulation. In the individual influence function case (when the size of the group is one), up-weighting a sample by ϵ leads to down-weighting other samples by $\epsilon/(m-1)$ whose effect can be neglected similar to the formulation of (Koh and Liang, 2017). In our formulation for the group influence function, we assume that the weights of samples in the set $\mathcal{U}$ has been up-weighted all by ϵ and use $p = \frac{|\mathcal{U}|}{|\mathcal{S}|}$ to denote the fraction of up-weighted training samples. This leads to a down-weighting of the rest of training samples by $\tilde{\epsilon} = \frac{|\mathcal{U}|}{|\mathcal{S}|-|\mathcal{U}|}\epsilon$, to preserve the empirical weight distributioxn of the training data. This is also important in order to have a fair comparison with the ground-truth leave-out-retraining estimates. Therefore, the resulting ERM can be written as:

$$\theta_{\mathcal{U}}^{\epsilon} = \arg\min_{\theta} L_{\mathcal{U}}^{\epsilon}(\theta)$$

where

$$L_{\mathcal{U}}^{\epsilon}(\theta) = \frac{1}{|\mathcal{S}|}\Big(\sum_{z\in\mathcal{S}\setminus\mathcal{U}}(1-\tilde{\epsilon})\ell(h_\theta(z)) + \sum_{z\in\mathcal{U}}(1+\epsilon)\ell(h_\theta(z))\Big). \tag{7}$$

Or equivalently In the above formulation, if $\epsilon = 0$ we get the original loss function $L_\emptyset(\theta)$ (where none of the training samples are removed) and if $\epsilon = -1$, we get the loss function $L_{\mathcal{U}}(\theta)$ (where samples are removed from training).

Let $\theta_{\mathcal{U}}^{\epsilon}$ denote the optimal parameters for $L_{\mathcal{U}}^{\epsilon}$ minimization. Essentially we are concerned about the change in the model parameters (i.e. $\Delta\theta = \theta_{\mathcal{U}}^{\epsilon} - \theta^*$) when each training sample in a group of size $|\mathcal{U}|$ is upweighted by a factor of ϵ. The key step of the derivation is to expand $\theta_{\mathcal{U}}^{\epsilon}$ around θ^* (the minimizer of $L_{\mathcal{U}}^{0}(\theta)$, or $L_\emptyset(\theta)$) with respect to the order of ϵ, the upweighting parameter. In order to do that, we use the perturbation theory (Avrachenkov, Filar, and Howlett, 2013) to expand $\theta_{\mathcal{U}}^{\epsilon}$ around θ^*.

Frequently used in quantum mechanics and also in other areas of physics such as particle physics, condensed matter and atomic physics, perturbation theory finds approximate solution to a problem ($\theta_{\mathcal{U}}^{\epsilon}$) by starting from the exact solution of a closely related and simpler problem (θ^*). As ϵ gets smaller and smaller, these higher order terms become less significant. However, for large model perturbations (such as the case of group influence functions), using higher-order terms can reduce approximation errors significantly. The following perturbation series forms the core of our derivation for second-order influence functions:

$$\theta_{\mathcal{U}}^{\epsilon} - \theta^* = \mathcal{O}(\epsilon)\theta^{(1)} + \mathcal{O}(\epsilon^2)\theta^{(2)} + \mathcal{O}(\epsilon^3)\theta^{(3)} + \cdots \tag{8}$$

where $\theta^{(1)}$ characterizes the first-order (in ϵ) perturbation vector of model parameters while $\theta^{(2)}$ is the second-order (in ϵ) model perturbation vector. We hide the dependencies of these perturbation vectors to constants (such as $|U|$) with the $\mathcal{O}(.)$ notation.

In the case of computing influence of individual points, as shown by (Koh and Liang, 2017), the scaling of $\theta^{(1)}$ is in the order of $1/|\mathcal{S}|$ while the scaling of the second-order coefficient is $1/|\mathcal{S}|^2$ which is very small when $\mathcal{S}$ is large. Thus, in this case, the second-order term can be ignored. In the case of computing the group influence, the second-order coefficient is in the order of $|\mathcal{U}|^2/|\mathcal{S}|^2$, which can be large when the size of $\mathcal{U}$ is large. Thus, in our definition of the group influence function, both $\theta^{(1)}$ and $\theta^{(2)}$ are taken into account.

The first-order group influence function (denoted by $\mathcal{I}^{(1)}$) when all the samples in a group $\mathcal{U}$ are up-weighted by ϵ can be defined as:

$$\mathcal{I}^{(1)}(\mathcal{U}) = \frac{\partial\theta_{\mathcal{U}}^{\epsilon}}{\partial\epsilon}|_{\epsilon=0}$$
$$= \frac{\partial(\theta^* + \mathcal{O}(\epsilon)\theta^{(1)} + \mathcal{O}(\epsilon^2)\theta^{(2)})}{\partial\epsilon}|_{\epsilon=0} = \theta^{(1)}$$

To capture the dependency of the terms in $\mathcal{O}(\epsilon^2)$, on the group influence function, we define $\mathcal{I}'$ as follows:

$$\mathcal{I}'(\mathcal{U}) = \frac{\partial^2\theta_{\mathcal{U}}^{\epsilon}}{\partial\epsilon^2}|_{\epsilon=0}$$
$$= \frac{\partial^2(\theta^* + \mathcal{O}(\epsilon)\theta^{(1)} + \mathcal{O}(\epsilon^2)\theta^{(2)})}{\partial\epsilon^2}|_{\epsilon=0} = \theta^{(2)}$$

Although one can consider even higher-order terms, in this paper, we restrict our derivations up to the second-order approximations of the group influence function. We now state our main result in the following theorem:

Theorem 1. *If the third-derivative of the loss function at θ^* is sufficiently small, the second-order group influence function (denoted by $\mathcal{I}^{(2)}(\mathcal{U})$) when all samples in a group $\mathcal{U}$ are up-weighted by ϵ is:*

$$\mathcal{I}^{(2)}(\mathcal{U}) = \mathcal{I}^{(1)}(\mathcal{U}) + \mathcal{I}'(\mathcal{U}) \tag{9}$$

where:

$$\mathcal{I}^{(1)}(\mathcal{U}) = -\frac{1}{1-p}\frac{1}{|\mathcal{S}|}H_{\theta^*}^{-1}\sum_{z\in\mathcal{U}}\nabla\ell(h_{\theta^*}(z))$$

and

$$\mathcal{I}'(\mathcal{U}) = \frac{p}{1-p}\Big(I - (\nabla^2 L_\emptyset(\theta^*))^{-1}\frac{1}{|\mathcal{U}|}\sum_{z\in\mathcal{U}}\nabla^2\ell(h_{\theta^*}(z))\Big)\theta^{(1)}$$

This result is based on the assumption that the third-order derivatives of the loss function at θ^* is small. For the quadratic loss, the third-order derivatives of the loss are zero. Our experiments with the cross-entropy loss function indicates that this assumption approximately holds for the classification problem as well. Below, we present a concise sketch of this result.

Proof Sketch. We now derive $\theta^{(1)}$ and $\theta^{(2)}$ to be used in the second order group influence function $\mathcal{I}^{(2)}(\mathcal{U})$. As $\theta_{\mathcal{U}}^{\epsilon}$ is the optimal parameter set for the interpolated loss function

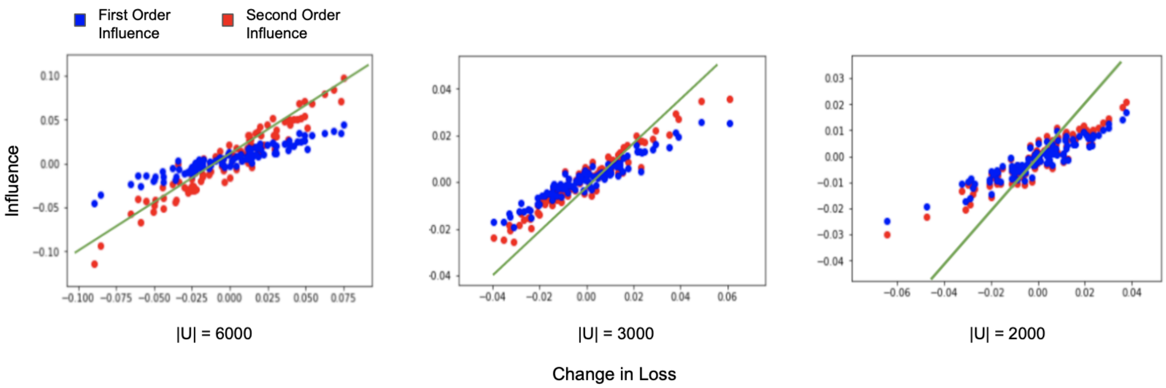

Figure 1. Comparison of first-order and second-order group influences in case of synthetic dataset with 10,000 samples using logistic regression for a mis-classified test point. Across different sizes of groups which were randomly selected, it can be observed that the second-order influence values are more correlated with the ground truth than that of the first-order ones. The green line highlights the $y = x$ line.

$L_{\mathcal{U}}^{\epsilon}(\theta)$, due to the first-order stationary condition, we have the following equality:

$$0 = \nabla L_{\mathcal{U}}^{\epsilon}(\theta_{\mathcal{U}}^{\epsilon}) = \nabla L_{\emptyset}(\theta_{\mathcal{U}}^{\epsilon}) + \frac{1}{|\mathcal{S}|}(-\tilde{\epsilon} \sum_{z \in \mathcal{S} \setminus \mathcal{U}} + \epsilon \sum_{z \in \mathcal{U}}) \nabla \ell(h_{\theta_{\mathcal{U}}^{\epsilon}}(z)) \tag{10}$$

The main idea is to use Taylor series for expanding $\nabla L_{\emptyset}(\theta_{\mathcal{U}}^{\epsilon})$ around θ^* along with the perturbation series defined in Equation (8) and compare the terms of the same order in ϵ:

$$\nabla L_{\emptyset}(\theta_{\mathcal{U}}^{\epsilon}) = \nabla L_{\emptyset}(\theta^*) + \nabla^2 L_{\emptyset}(\theta^*)(\theta_{\mathcal{U}}^{\epsilon} - \theta^*) + \dots \tag{11}$$

Similarly, we expand $\nabla \ell(h_{\theta_{\mathcal{U}}^{\epsilon}}(z))$ around θ^* using Taylor series expansion. To derive $\theta^{(1)}$ we compared terms with the coefficient of $\mathcal{O}(\epsilon)$ in Equation (10) and for $\theta^{(2)}$ we compared terms with coefficient $\mathcal{O}(\epsilon^2)$. Based on this, $\theta^{(1)}$ can be written in the following way:

$$\theta^{(1)} = -\frac{1}{1-p}\frac{1}{|\mathcal{S}|} H_{\theta^*}^{-1} \sum_{z \in \mathcal{U}} \nabla \ell(h_{\theta^*}(z)) \tag{12}$$

We expand Equation(10) and compare the terms with coefficient $\mathcal{O}(\epsilon)$:

$$\begin{aligned} &\epsilon \nabla^2 L_{\emptyset}(\theta^*)\theta^{(1)} \\ &= \frac{1}{|\mathcal{S}|}(\tilde{\epsilon} \sum_{z \in \mathcal{S} \setminus \mathcal{U}} - \epsilon \sum_{z \in \mathcal{U}}) \nabla \ell(h_{\theta^*}(z)) \\ &= \tilde{\epsilon} \nabla L_{\emptyset}(\theta^*) - \frac{1}{|\mathcal{S}|}(\tilde{\epsilon} + \epsilon) \sum_{z \in \mathcal{U}} \nabla \ell(h_{\theta^*}(z)) \\ &= -\frac{1}{|\mathcal{S}|}(\tilde{\epsilon} + \epsilon) \sum_{z \in \mathcal{U}} \nabla \ell(h_{\theta^*}(z)) \\ &= -\frac{1}{|\mathcal{S}|}\frac{1}{(1-p)} \epsilon \sum_{z \in \mathcal{U}} \nabla \ell(h_{\theta^*}(z)) \end{aligned} \tag{13}$$

$\theta^{(1)}$ is the first-order approximation of group influence function and can be denoted by $\mathcal{I}^{(1)}$. Note that our first-order approximation of group influence function $\mathcal{I}^{(1)}$, is slightly different from (Koh et al., 2019a) with an additional $1 - p$ in the denominator. For $\theta^{(2)}$ we compare the terms with coefficients of the same order of $\mathcal{O}(\epsilon^2)$ in Equation (10):

$$\begin{aligned} &\epsilon^2 \nabla^2 L_{\emptyset}(\theta^*)\theta^{(2)} + \frac{1}{2} L_{\emptyset}^{'''}(\theta^*)[\epsilon\theta^{(1)}, \epsilon\theta^{(1)}, I] \\ &+ \frac{1}{|\mathcal{S}|}(-\tilde{\epsilon} \sum_{\mathcal{S} \setminus \mathcal{U}} + \epsilon \sum_{\mathcal{U}}) \nabla^2 \ell(h_{\theta^*}(z))(\epsilon\theta^{(1)}) \\ &= 0 \end{aligned} \tag{14}$$

For the $\theta^{(2)}$ term, we ignore the third-order term $\frac{1}{2} L_{\emptyset}^{'''}(\theta^*)[\epsilon\theta^{(1)}, \epsilon\theta^{(1)}, I]$ due to it being small. Now we substitute the value of $\tilde{\epsilon}$ and equate the terms with coefficient in the order of $\mathcal{O}(\epsilon^2)$:

$$\begin{aligned} \nabla^2 L_{\emptyset}(\theta^*)\theta^{(2)} = \frac{|\mathcal{U}|}{|\mathcal{S}| - |\mathcal{U}|} \Big(&\frac{1}{|\mathcal{S}|} \sum_{z \in \mathcal{S}} \nabla^2 \ell(h_{\theta^*}(z)) \\ &- \frac{1}{|\mathcal{U}|} \sum_{z \in \mathcal{U}} \nabla^2 \ell(h_{\theta^*}(z)) \Big) \theta^{(1)} \end{aligned} \tag{15}$$

Rearranging the Equation (15), we get the same identity as $\mathcal{I}^{'}$ in Theorem (1). □

It can be observed that the additional term ($\mathcal{I}^{'}$) in our second-order approximation captures cross-dependencies among the samples in $\mathcal{U}$ through a function of gradients and Hessians of the loss function at the optimal model parameters. This makes the second-order group influence function to be more informative when training samples are correlated. In Section

(7), we empirically show that the addition of $\mathcal{I}^{'}$ improves correlation with the ground truth influence as well.

For tracking the change in the test loss for a particular test point z_t when a group $\mathcal{U}$ is removed, we use the chain rule to compute the influence score as follows:

$$\mathcal{I}^{(2)}(\mathcal{U}, z_t) = \nabla \ell(h_{\theta^*}(z_t))^T \left(\mathcal{I}^{(1)}(\mathcal{U}) + \mathcal{I}^{'}(\mathcal{U})\right) \quad (16)$$

Our second-order approximation of group influence function consists of a first-order term that is similar to the one proposed in (Koh et al., 2019a) with an additional scaling term $1/(1-p)$. This scaling is due to the fact that our formulation preserves the empirical weight distribution constraint in ERM, which is essential when a large group is up-weighted. The second-order influence function has an additional term $\mathcal{I}^{'}$ that is directly proportional to p and captures large perturbations to the model parameters more effectively.

5. Selection of Influential Groups

In this section, we explain how the second-order group influence function can be used to select the most influential group of training samples for a particular test prediction. In case of the existing first-order approximations for group influence functions, selecting the most influential group can be done greedily by ranking the training points with the highest individual influence since the group influence is the sum of influence of the individual points. However, with the second-order approximations such greedy selection is not optimal since the group influence is not additive in terms of the influence of individual points. To deal with this issue, we first decompose the second-order group influence function $\mathcal{I}^{(2)}(\mathcal{U}, z_t)$ into two terms as:

$$\nabla \ell(h_{\theta^*}(z_t))^T \Bigg\{ \underbrace{\frac{1}{|\mathcal{S}|} \frac{1-2p}{(1-p)^2} H_{\theta^*}^{-1} \sum_{z \in \mathcal{U}} \nabla \ell(h_{\theta^*}(z)) +}_{Term1}$$

$$\underbrace{\frac{1}{(1-p)^2} \frac{1}{|\mathcal{S}|^2} \sum_{z \in \mathcal{U}} H_{\theta^*}^{-1} \nabla^2 \ell(h_{\theta^*}(z)) H_{\theta^*}^{-1} \sum_{z' \in \mathcal{U}} \nabla \ell(h_{\theta^*}(z')) \Bigg\}}_{Term2} \quad (17)$$

where $H_{\theta^*} = \nabla^2 L_{\emptyset}(\theta^*)$. While $Term1$ is additive with respect to the samples and $Term2$ has pairwise dependencies among samples.

To simplify notation, we define the constant vector $\nabla \ell(h_{\theta^*})(z_t)^T H_{\theta^*}^{-1}$ as v_1. Ideally for a given fixed group of size k, we want to find k training samples amongst the total m training samples which maximizes the influence for a given test point z_t. We can define this in the form of a quadratic optimization problem as follows:

$$\max_{w} \quad c_1 w^T a + c_2 w^T B w \qquad (18)$$
$$\text{s.t.} \quad \|w\|_0 \leq k$$

where B is composed of two matrices C and D i.e. $B = CD$. w contains the weights associated with each sample in the training set. The entries of a contain $v_1^T \nabla \ell(h_{\theta^*}(z_i)) \; \forall i \in [1, m]$ and the rows of C contain $v_1^T \nabla^2 \ell(h_{\theta^*}(z_i)) H_{\theta^*}^{-1} \; \forall i \in [1, m]$. In case of D, the columns contain $\nabla \ell(h_{\theta^*}(z_i)) \; \forall i \in [1, m]$. We define the constant $\frac{1}{|\mathcal{S}|} \frac{1-2p}{(1-p)^2}$ as c_1 and $\frac{1}{(1-p)^2} \frac{1}{|\mathcal{S}|^2}$ as c_2.

This optimization can be relaxed using the $L_0 - L_1$ relaxation as done in applications of compressed sensing (Donoho, 2006; Candes and Tao, 2005; Ramirez, 2013). The relaxed optimization can then be solved efficiently using the projected gradient descent as denoted in (Liu and Ye, 2009; Duchi, Shalev-Shwartz, Singer, and Chandra, 2008).

6. Computational Complexity

For models with a relatively large number of parameters, computing the inverse of the Hessian $H_{\theta^*}^{-1}$ can be expensive and is of the order of $O(n^3)$. However, computing the Hessian-vector product (Pearlmutter, 1994) is relatively computationally inexpensive. In our experiments similar to (Koh and Liang, 2017; Koh et al., 2019a; Chen et al., 2020), we used conjugate gradients (a second-order optimization technique) (Shewchuk, 1994) to compute the inverse Hessian-vector product which uses a Hessian-vector product in the routine thus saving the expense for inverting the Hessian directly. The proposed second-order group influence function can be computed similarly to the first-order group influence functions with only an additional step of Hessian-vector product.

7. Experiments

7.1. Setup

Our goal through the experiments is to observe if the second-order approximations of group influence functions improve the correlation with the ground truth estimate across different settings. We compare the computed second-order group influence score with the ground truth influence (which is computed by leave-k-out retraining for a group with size k). Our metric for evaluation is the Pearson correlation which measures how linearly the computed influence and the actual ground truth estimate are related. We perform our experiments primarily on logistic regression where the group influence function is well-defined. Additionally we also check the accuracy of first-order and second-order group influence functions in case of neural networks.

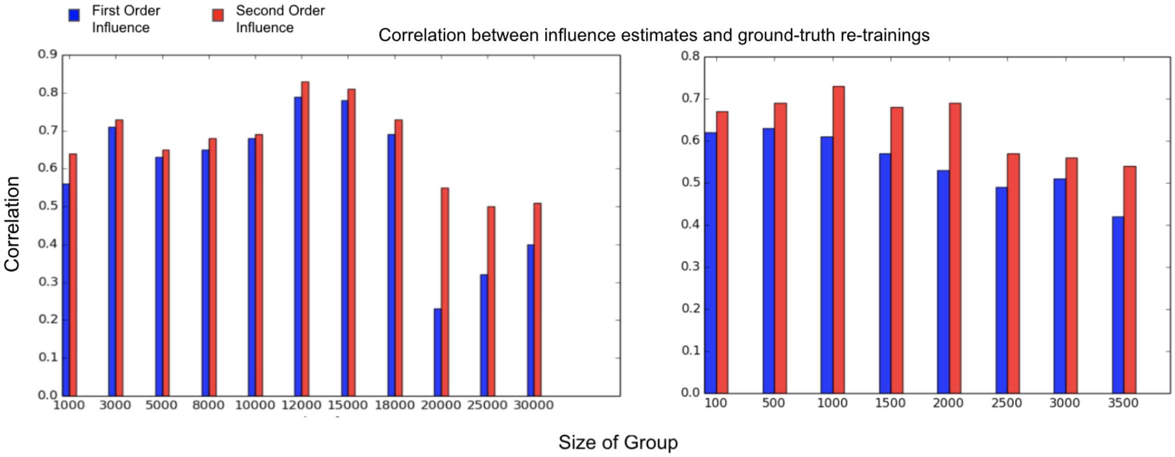

Figure 2. Group size vs the correlation with the ground truth on MNIST for logistic regression with random groups (left panel) and *coherent* groups (right panel).

7.2. Datasets

To understand the accuracy of both first-order and second-order group influence functions on linear models we use two datasets. In our first experiments, we use a synthetic dataset along with logistic regression. The synthetic dataset has 10,000 points drawn from a Gaussian distribution, consisting of 5 features and 2 classes. The details for the synthetic data can be found in the Appendix. The second set of experiments are done with the standard handwritten digits database MNIST (LeCun, Bottou, Bengio, and Haffner, 1998) which consists of 10 classes of different digits. For understanding how group influence functions behave in case of the neural networks we use the MNIST dataset. For each of the two datasets, we pick random groups as well *coherent* groups as in (Koh et al., 2019a) with sizes ranging from 1.6% to 60% of the entire training points. The computed group influence was primarily investigated for a test-point which was misclassified by the model. A detailed description of how the groups were selected in our experiments is given in the Appendix. For the optimal group selection we used a synthetic dataset consisting of 20,000 training points consisting of 5 features in the form of 4 isotropic Gaussian blobs.

7.3. Observations and Analysis

7.3.1. Linear Models

For logistic regression, the general observation for the randomly selected groups was that the second-order group influence function improves the correlation with the ground truth estimates across different group sizes in both the synthetic dataset as well as MNIST. For the synthetic dataset, in Figure (1), it can be observed that the approximation provided by the second-order group influence function is fairly close to the ground truth when a large fraction of the training data (60 $\%$) is removed. In such cases of large group sizes, the first-order approximation of group influence function is relatively inaccurate and far from the ground truth influence. This observation is consistent with the small perturbation assumption of first-order influence functions. However, in cases of smaller group sizes, although the second-order approximation improves over existing first-order group influence function, the gain in correlation is small. In case of MNIST, the observation was similar where the gain in correlation was significant when the size of the considered group was large. For e.g. it can seen in Figure (2), that when more than 36% of the samples were removed, the gain in correlation is almost always more than 40%. While the improvement in correlation for larger group sizes is consistent with our theory that the second-order approximation is effective in the case of large changes to the model, the gain in correlation is non-monotonic with respect to the group sizes. For groups of small size, selected uniformly at random, the model parameters do not change significantly and the second-order approximation improves only marginally over the existing first-order approximation. However, when a *coherent* group (a group having training examples from the same class) of even a relatively small size is removed, the perturbation to the model is larger (as the model parameters can change significantly in a particular direction) than if a random group is removed. In such settings, we observe that even for small group sizes, the second-order approximation consistently improves the correlation with the ground-truth significantly (Figure (2)). For *coherent* groups, across different group sizes of the MNIST dataset, we observed an improvement in correlation when the second-order approximation was used. Across different group sizes we observed that the gain in correlation is at least 15%. These observations

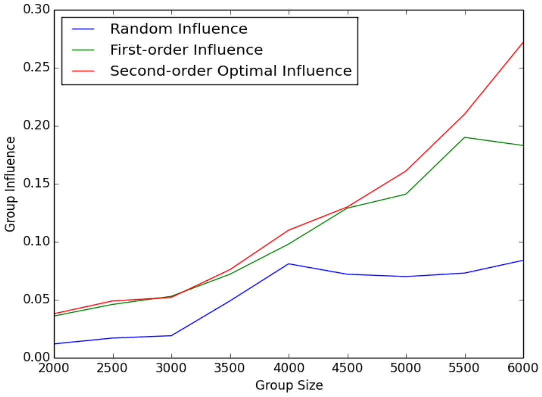

Figure 3. Optimal group selection on synthetic data.

(shown in Figure (2)) reinforces our theory that the second-order (or rather higher-order) approximations of influence functions are particularly effective when the perturbation or changes in the model parameters are significantly large. The second-order approximation of the influence function could thus be used over existing first-order approximations in practical purposes such as understanding the behaviour of training groups with similar properties (e.g. demographic groups) on model predictions, without the need to actually retrain the model again.

7.3.2. Neural Networks

In case of neural networks, the Hessian is not positive semi-definite in general, which violates the assumptions of influence functions. Previously (Koh and Liang, 2017) regularized the Hessian in the form of $H_{\theta^*} + \lambda I$, and had shown that for the top few influential training points (not groups) and for a given test point, the correlation with the ground truth influence is still satisfactory, if not highly significant. However, how influence functions behave in the case of groups, is a topic not yet well explored. For MNIST, we used a regularized Hessian with a value of $\lambda = 0.01$ and conducted experiments for a relatively simple two hidden layered feed-forward network with sigmoid activations for both first-order and second-order group influence functions. The general observation was that both existing first and proposed second-order group influence functions underestimate the ground truth influence values across different group sizes, leading to a non-significant correlation. The corresponding Figure can be referred to in the Appendix. However, we observed that while the second-order influence values still suffer from the underestimation issue, they improve the correlation marginally across different group sizes. This observation was consistent in cases of both random and *coherent* group selections.

7.3.3. Influential Group Selection

In order to validate the selection of the most influential group through the second-order approximation of influence function, we performed an experiment with logistic regression (where both first-order and second-order influence function estimates are fairly close to the ground truth) on a synthetic dataset. Across different group sizes we compared the group influence (through the second-order approximation and computed with Equation (18)) with the first-order influence computed greedily for a particular group size and the mean influence of randomly selected groups across 100 group sampling iterations. In our experiments we relaxed the L_0 norm to L_1 norm and solved the projected gradient descent step of the optimization in Equation (18) using (Duchi et al., 2008). We observed that the optimal group selection procedure led to groups having relatively higher influence computed with the second-order approximation when compared to the greedy first-order influence and randomly selected groups corresponding to the different group-sizes ranging from 10% to 30% of the total training samples. Specifically the optimal group influence was significantly higher than the greedy first-order group influence when the group sizes were relatively large. The selection procedure could be practically used to detect the most relevant subset of training examples which impacts a particular test-time decision through a given machine learning model when the second-order influence function is used.

8. Conclusion and Future Work

In this paper, we proposed second-order group influence functions for approximating model changes when a group from the training set is removed. Empirically, in the case of linear models, across different group sizes and types, we showed that the second-order influence has a higher correlation with ground truth values compared to the first-order ones and is more effective than existing first-order approximations. Our observation was that the second-order influence is significantly informative when the changes to the underlying model is relatively large. We showed that the proposed second-order group influence function can be practically used in conjunction with optimization techniques to select the most influential group in the training set for a particular test prediction. For non-linear models such as deep neural networks, we observed that both first-order and second-order influence functions lead to a non-significant correlation with the ground truth across different group sizes (although the correction values for the second-order method was marginally better). Developing accurate group influence functions for neural networks or training neural networks to have improved influence functions and also extending group influence functions to the transfer learning setting as in (Chen et al., 2020) are among directions for future work.

9. Acknowledgements

This project was supported in part by NSF CAREER AWARD 1942230, HR001119S0026-GARD-FP-052, AWS Machine Learning Research Award, a sponsorship from Capital One, and Simons Fellowship on “Foundations of Deep Learning”.

References

A. Alaa and M. Van Der Schaar. Validating causal inference models via influence functions. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 191–201, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/alaa19a.html.

I. Arrieta-Ibarra, L. Goff, D. Jiménez-Hernández, J. Lanier, and E. G. Weyl. Should we treat data as labor? moving beyond "free". *AEA Papers and Proceedings*, 108:38–42, May 2018. doi: 10.1257/pandp.20181003. URL http://www.aeaweb.org/articles?id=10.1257/pandp.20181003.

K. E. Avrachenkov, J. A. Filar, and P. G. Howlett. *Analytic Perturbation Theory and Its Applications*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2013. ISBN 1611973139, 9781611973136.

M. Brunet, C. Alkalay-Houlihan, A. Anderson, and R. S. Zemel. Understanding the origins of bias in word embeddings. *CoRR*, abs/1810.03611, 2018. URL http://arxiv.org/abs/1810.03611.

E. J. Candes and T. Tao. Decoding by linear programming. *IEEE Trans. Inf. Theor.*, 51(12):4203–4215, Dec. 2005. ISSN 0018-9448. doi: 10.1109/TIT.2005.858979. URL http://dx.doi.org/10.1109/TIT.2005.858979.

H. Chen, S. Si, Y. Li, C. Chelba, S. Kumar, D. Boning, and C.-J. Hsieh. {MULTI}-{stage} {influence} {function}, 2020. URL https://openreview.net/forum?id=r1geR1BKPr.

I. Chen, F. D. Johansson, and D. Sontag. Why is my classifier discriminatory? In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 3539–3550. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/7613-why-is-my-classifier-discriminatory.pdf.

R. Cook and S. Weisberg. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 22(4):495–508, 1980. ISSN 0040-1706. doi: 10.1080/00401706.1980.10486199.

R. D. Cook and W. Sanford. Residuals and influence in regression. *Chapman and Hall*, 1982. URL http://hdl.handle.net/11299/37076.

D. L. Donoho. Compressed sensing. *IEEE Trans. Inf. Theor.*, 52(4):1289–1306, Apr. 2006. ISSN 0018-9448. doi: 10.1109/TIT.2006.871582. URL https://doi.org/10.1109/TIT.2006.871582.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, ICML ’08, pages 272–279, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390191. URL http://doi.acm.org/10.1145/1390156.1390191.

B. Efron. Jackknife-after-bootstrap standard errors and influence functions. *Journal of the Royal Statistical Society. Series B (Methodological)*, 54(1):83–127, 1992. ISSN 00359246. URL http://www.jstor.org/stable/2345949.

R. Giordano, M. I. Jordan, and T. Broderick. A higher-order swiss army infinitesimal jackknife. *ArXiv*, abs/1907.12116, 2019.

J. Hayes and O. Ohrimenko. Contamination attacks and mitigation in multi-party machine learning. *CoRR*, abs/1901.02402, 2019. URL http://arxiv.org/abs/1901.02402.

L. A. Jaeckel. The infinitesimal jackknife. *Technical Report*, 1:1–35, 06 1972.

R. James, L. Lingling, T. Eric, and A. van der Vaart. Higher order influence functions and minimax estimation of nonlinear functionals. *Probability and Statistics: Essays in Honor of David A. Freedman, 335–421*, abs/1706.03825, 2017. URL https://projecteuclid.org/euclid.imsc/1207580092.

P. W. Koh and P. Liang. Understanding black-box predictions via influence functions. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/koh17a.html.

P. W. Koh, K. Ang, H. H. K. Teo, and P. Liang. On the accuracy of influence functions for measuring group effects. *CoRR*, abs/1905.13289, 2019a. URL http://arxiv.org/abs/1905.13289.

P. W. Koh, J. Steinhardt, and P. Liang. Stronger data poisoning attacks break data sanitization defenses. *arXiv preprint arXiv:1811.00741*, 2019b.

P. Langley. Crafting papers on machine learning. In P. Langley, editor, *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pages 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, volume 86, pages 2278–2324, 1998. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.42.7665.

W.-Y. Lin, Y.-H. Hu, and C.-F. Tsai. Machine learning in financial crisis prediction: A survey. *Trans. Sys. Man Cyber Part C*, 42(4):421–436, July 2012. ISSN 1094-6977. doi: 10.1109/TSMCC.2011.2170420. URL https://doi.org/10.1109/TSMCC.2011.2170420.

J. Liu and J. Ye. Efficient euclidean projections in linear time. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 657–664, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553459. URL http://doi.acm.org/10.1145/1553374.1553459.

A. S. Lundervold and A. Lundervold. An overview of deep learning in medical imaging focusing on MRI. *CoRR*, abs/1811.10052, 2018. URL http://arxiv.org/abs/1811.10052.

D. Madras, J. Atwood, and A. D'Amour. Detecting extrapolation with influence functions. *ICML Workshop*, 2019.

B. A. Pearlmutter. Fast exact multiplication by the hessian. *Neural Comput.*, 6(1):147–160, Jan. 1994. ISSN 0899-7667. doi: 10.1162/neco.1994.6.1.147. URL http://dx.doi.org/10.1162/neco.1994.6.1.147.

C. Ramirez. Why l1 is a good approximation to l0: A geometric explanation. *Journal of Uncertain Systems*, 7:203–207, 08 2013.

P. G. Schulam and S. Saria. Can you trust this prediction? auditing pointwise reliability after learning. In *AISTATS*, 2019.

F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, Mar. 2002. ISSN 0360-0300. doi: 10.1145/505282.505283. URL http://doi.acm.org/10.1145/505282.505283.

J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. -, 1994.

R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer-Verlag, Berlin, Heidelberg, 1st edition, 2010. ISBN 1848829345, 9781848829343.

H. Wang, B. Ustun, and F. P. Calmon. Repairing without retraining: Avoiding disparate impact with counterfactual distributions. *CoRR*, abs/1901.10501, 2019. URL http://arxiv.org/abs/1901.10501.

Y. Yang, G. Li, H. Qian, K. C. Wilhelmsen, Y. Shen, and Y. Li. Smnn: Batch effect correction for single-cell rna-seq data via supervised mutual nearest neighbor detection. *bioRxiv*, 2019. doi: 10.1101/672261. URL https://www.biorxiv.org/content/early/2019/06/17/672261.