Understanding Black-box Predictions with Influence Functions

Pang Wei Koh
Percy Liang
Top-5 error on ImageNet

- Computer vision
- Deep learning
- Human

[1] Defense Systems Information Analysis Center
Top-5 error on ImageNet

[1] Defense Systems Information Analysis Center
Given a high-accuracy, black-box model, and a prediction from it, can we answer...
Why did the model make this prediction?
Why did the model make this prediction?

- Make better decisions [1]
- Improve the model [2]
- Discover new science [3]
- Provide end-users explanations [4]

What inputs maximally activate these neurons? [1]

Can we represent this model with a simpler one? [2-3, 9]

Which part of the input was most responsible for this prediction? [4-9]

[1] Girshick et al., 2014
[6] Li, Monroe, and Jurafsky, 2016
[8] Sundararajan, Taly, and Yan, 2017
[9] Leino et al., 2018
Training data

“Dog”

Fish

Dog

Dog
Why did the model make this prediction?

Which training points were most responsible for this prediction?
Fish

Dog

Dog

Training data $z_1, z_2, \ldots, z_n$
Pick $\hat{\theta}$ to minimize $\frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$.
Pick $\hat{\theta}$ to minimize $\frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$

Training data $z_1, z_2, \ldots, z_n$
Pick $\hat{\theta}$ to minimize $\frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$.
Pick $\hat{\theta}$ to minimize $\frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$

Pick $\hat{\theta}_{\epsilon,z_{\text{train}}}$ to minimize

$$\frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z_{\text{train}}, \theta)$$

Training data $z_1, z_2, \ldots, z_n$

“Dog”
"Dog" (79% confidence) vs. "Dog" (82% confidence)

\[ \hat{\theta} \]

\[ \hat{\theta}_{\epsilon, z_{\text{train}}} \]

Test input
"Dog" (79% confidence) vs. "Dog" (82% confidence)

What is $L(z_{\text{test}}, \hat{\theta}_{\epsilon, z_{\text{train}}}) - L(z_{\text{test}}, \hat{\theta})$?
Why did the model make this prediction?

Which training points were most responsible for this prediction?

How would the prediction change if we upweighted each training point?
Motivation

Influence functions

Applications

Conclusion
Influence functions

- Introduced in the 1970s in the field of robust statistics (e.g., Jaeckel, 1972; Cook, 1977; Cook and Weisberg, 1982)
Influence functions

- Introduced in the 1970s in the field of robust statistics (e.g., Jaeckel, 1972; Cook, 1977; Cook and Weisberg, 1982)
- Consider an estimator $T$ that acts on a distribution $F$
- How much does $T$ change if we perturb $F$?
Influence functions

- Goal: Measure change in $L(z_{\text{test}}, \hat{\theta}_\epsilon, z_{\text{train}})$ as we increase $\epsilon$. 

Under smoothness assumptions, $I_{\text{up,loss}}(z_{\text{train}}, z_{\text{test}})$ is defined as:

$$dL(z_{\text{test}}, \hat{\theta}_\epsilon, z_{\text{train}}) = 0 = L(z_{\text{test}}, \hat{\theta}_\epsilon) + H_{\hat{\theta}_\epsilon} L(z_{\text{train}}, \hat{\theta}_\epsilon),$$

where $H_{\hat{\theta}_\epsilon}$ is defined as:

$$H_{\hat{\theta}_\epsilon} = \frac{1}{n} \sum_{i=1}^{n} 2L(z_i, \hat{\theta}_\epsilon).$$
Influence functions

- Goal: Measure change in $L(z_{test}, \hat{\theta}_\epsilon, z_{train})$ as we increase $\epsilon$.

- $\hat{\theta}_{\epsilon, z_{train}} \overset{\text{def}}{=} \arg \min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z_{train}, \theta)$. 
Influence functions

- Goal: Measure change in $L(z_{\text{test}}, \hat{\theta}_\epsilon, z_{\text{train}})$ as we increase $\epsilon$.

- $\hat{\theta}_{\epsilon, z_{\text{train}}} \overset{\text{def}}{=} \arg \min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z_{\text{train}}, \theta)$.

- Under smoothness assumptions,

$$
\mathcal{I}_{\text{up, loss}}(z_{\text{train}}, z_{\text{test}}) \overset{\text{def}}{=} \frac{dL(z_{\text{test}}, \hat{\theta}_{\epsilon, z_{\text{train}}})}{d\epsilon} \bigg|_{\epsilon=0}
$$
Influence functions

• Goal: Measure change in $L(z_{\text{test}}, \hat{\theta}_\epsilon, z_{\text{train}})$ as we increase $\epsilon$.

• $\hat{\theta}_{\epsilon, z_{\text{train}}} \overset{\text{def}}{=} \arg \min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z_{\text{train}}, \theta)$.

• Under smoothness assumptions,

$$I_{\text{up,loss}}(z_{\text{train}}, z_{\text{test}}) \overset{\text{def}}{=} \left. \frac{dL(z_{\text{test}}, \hat{\theta}_\epsilon, z_{\text{train}})}{d\epsilon} \right|_{\epsilon=0} = -\nabla_{\theta} L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_{\theta} L(z_{\text{train}}, \hat{\theta}),$$

• where $H_{\hat{\theta}} \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \nabla^2_{\theta} L(z_i, \hat{\theta})$. 
RBF SVM (raw pixels)  Logistic regression (Inception features)

Test image
Most influential
Most harmful
Potential issues

*More details in paper
Potential issues

1. Computational inefficiency
Potential issues

1. Computational inefficiency

\[
\begin{bmatrix}
H
\end{bmatrix}^{-1}
\]

Slow

[1] Pearlmutter, 1994
Potential issues

1. Computational inefficiency

\[
H \quad V \\
\quad -1
\]

Fast [1]  Slow

[1] Pearlmutter, 1994
Potential issues

1. Computational inefficiency

\[
\begin{bmatrix}
H & \vdots \\
\vdots & \vdots \\
\end{bmatrix}
\begin{bmatrix}
v \\
\end{bmatrix}
\approx
\begin{bmatrix}
H & \vdots \\
\vdots & \vdots \\
\end{bmatrix}^{-1}
\begin{bmatrix}
v \\
\end{bmatrix}
\]

Fast [1]

[1] Pearlmutter, 1994
Potential issues

1. Computational inefficiency

2. Non-smooth losses
Potential issues

1. Computational inefficiency

2. Non-smooth losses
Potential issues

1. Computational inefficiency

2. Non-smooth losses

3. Difficulty in finding the global minimizer
Potential issues

1. Computational inefficiency
2. Non-smooth losses
3. Difficulty in finding the global minimizer

\[ \hat{\theta} \]

\[ \tilde{\theta} \] stuck here
Potential issues

1. Computational inefficiency
2. Non-smooth losses
3. Difficulty in finding the global minimizer
Potential issues

For a fixed $z_{\text{test}}$ and for each $z_{\text{train}}$, compared:

1. Actual change in $L(z_{\text{test}})$ after removing $z_{\text{train}}$

2. Predicted change in $L(z_{\text{test}})$ after removing $z_{\text{train}}$
Motivation

Influence functions

> Applications

Conclusion
Application 1
Debugging model errors
Debugging model errors

• If a model makes a mistake, can we find out why?

• Case study: hospital re-admission (logistic regression, 20K patients, 127 features)
Debugging model errors

Healthy + re-admitted adults

Healthy children

Re-admitted children

<table>
<thead>
<tr>
<th>Original</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>~20k</td>
<td>~20k</td>
</tr>
<tr>
<td>21</td>
<td>-20</td>
</tr>
<tr>
<td>3</td>
<td>same</td>
</tr>
</tbody>
</table>
Debugging model errors

True test label: Healthy
Model predicts: Re-admitted
Debugging model errors

True test label: Healthy
Model predicts: Re-admitted

Indicator for ‘child’

Feature weight

Top 20 features
Debugging model errors

True test label: Healthy
Model predicts: Re-admitted

Top 20 influential training examples

Influence

Healthy child

Re-admitted children
Debugging model errors

True test label: Healthy
Model predicts: Re-admitted

Top 20 features

Indicator for ‘child’
Application 2
Fixing training data
Fixing training data

- Setup: training labels are noisy, and we have a small budget to manually inspect them.
- Can we prioritize which labels to try to fix?
Fixing training data

- Setup: training labels are noisy, and we have a small budget to manually inspect them.
- Can we prioritize which labels to try to fix?

- Setup: training labels are noisy, and we have a small budget to manually inspect them.
- Can we prioritize which labels to try to fix?

<table>
<thead>
<tr>
<th>Ham</th>
<th>Spam</th>
<th>Spam</th>
<th>Ham</th>
<th>Ham</th>
<th>Spam</th>
</tr>
</thead>
</table>
Fixing training data

- Setup: training labels are noisy, and we have a small budget to manually inspect them
- Can we prioritize which labels to try to fix?

![Illustration of email classification with some labels crossed out.]

- Ham
- Spam
- Ham
- Spam
- Ham
- Spam
Fixing training data

- Key idea: if a training point is not influential, don’t waste effort checking it
Fixing training data

![Graph showing test accuracy vs fraction of train data checked for different data conditions: clean data, influence, loss, random. The graph illustrates how the test accuracy improves as more of the train data is checked, with the clean data line reaching the highest accuracy.]
Application 3
Adversarial training examples
Test data → Model → Correct prediction
Test data → Model → Wrong prediction
"panda" 57.7% confidence + .007 × = "gibbon" 99.3% confidence
Adversarial test examples

“panda”  57.7% confidence

+ .007 ×

= “gibbon”  99.3% confidence
Adversarial test examples
Follow the gradient of the test loss w.r.t. test features (to increase loss) [1]

We have adversarial test examples. Can we create adversarial \textit{training} examples?
Adversarial training examples
Follow the gradient of the test loss w.r.t. train features
Model

Test data (unmodified)

Training data

Gradient

Wrong prediction
Adversarial training examples
Follow the gradient of the test loss w.r.t. train features
Influence functions help us calculate this gradient
Adversarial training examples
Follow the gradient of the test loss w.r.t. train features
Influence functions help us calculate this gradient

*Mathematically equivalent to gradient-based attacks explored by Biggio et al. (2012), Mei & Zhu (2015), and others
Adversarial training examples

• Setup: dog vs. fish classification, logistic regression on top of Inception features
Adversarial training examples

A small perturbation to one training example:
Adversarial training examples

A small perturbation to one training example:

Can change multiple test predictions:

Orig (confidence): Dog (97%)
New (confidence): Fish (97%)

Dog (98%)  Fish (93%)
Dog (98%)  Fish (87%)
Dog (99%)  Fish (63%)
Dog (98%)  Fish (52%)
Three observations

1. Ambiguous examples are good attack vectors

Label: fish
Three observations

1. Ambiguous examples are good attack vectors
Three observations

1. Ambiguous examples are good attack vectors

2. Small change in pixels but large change in feature space
Three observations

1. Ambiguous examples are good attack vectors

2. Small change in pixels but large change in feature space

3. Attack makes model overfit to specific test examples ($n \sim d$)
Three observations

1. Ambiguous examples are good attack vectors

2. Small change in pixels but large change in feature space

3. Attack makes model overfit to specific test examples (n ~ d)
Outside world $\rightarrow$ Training data $\rightarrow$ Model $\rightarrow$ Wrong prediction $\rightarrow$ Test data.
Test data → Training data → Model → Correct prediction

Biggio et al., Poisoning attacks against support vector machines, 2012
Xiao et al., Is feature selection secure against training data poisoning?, 2015
Mei and Zhu, Using machine teaching to identify optimal training-set attacks on machine learners, 2015
Mozaffari-Kermani et al., Systematic poisoning attacks on and defenses for machine learning in healthcare, 2015
Burkard and Lagesse, Analysis of causative attacks against SVMs learning from data streams, 2017

Cretu et al., Casting out demons: Sanitizing training data for anomaly sensors, 2008
Rubinstein et al., Antidote: Understanding and defending against poisoning of anomaly detectors, 2009
Laishram and Phoha, Curie: A method for protecting SVM classifier from poisoning attack, 2016
Chen, He, and Hsu, Chen, He, and Hsu, Data sanitization against adversarial label contamination based on data complexity, 2017

...
Given a defense and a dataset, can we bound the damage that any attacker can do?
Motivation

Influence functions

Applications

> Conclusion
Why did the model make this prediction?

Which training points were most responsible for this prediction?

How would the prediction change if we upweighted each training point?
Dog Training

Training data

Fish

“Dog”

Training

Gradient
Future work

- Real-world problems: hospitals (interpretability, uncertainty)
- Real-world models: scale [1], non-convexity, SGD
- Studying global perturbations
- Connections to reliability and privacy [2]
- Influence as part of the objective [3]

Thank you


pangwei@cs.stanford.edu

Percy Liang

Koda

This presentation uses images from the Noun Project:

Question by Valeriy | Box by Rockicon | Magnifying Glass by il Capitano | services by IconsGhost | Ghost by Bakunetsu Kaito
Neural Network by Knut M. Synstad | Poisoned Dagger by Ben Davis | world by Aleksandr Vector | Shield by Nikita Kozin