Representer Point Selection for Explaining Deep Neural Networks

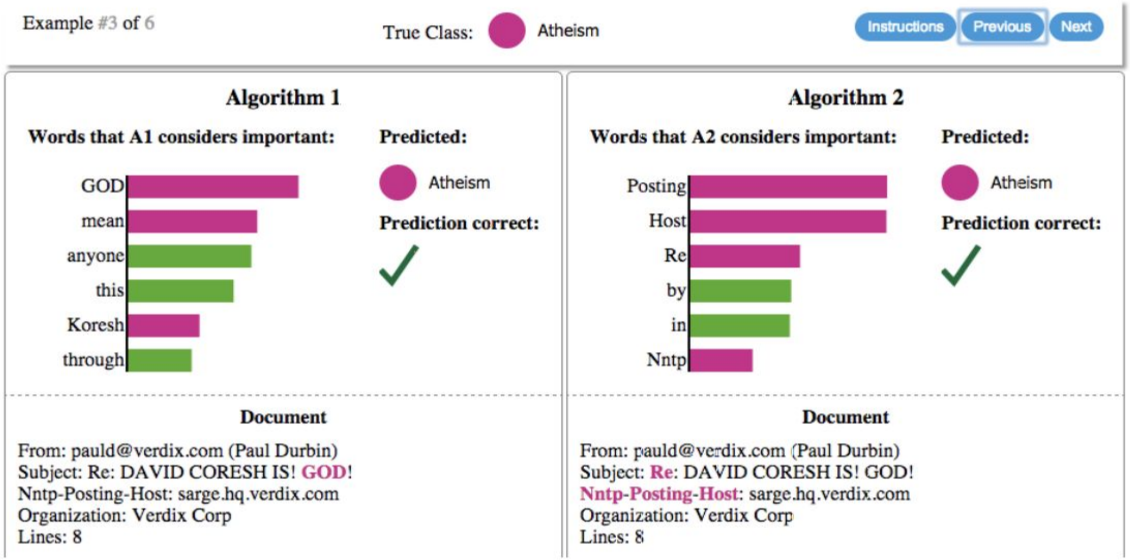
Chih-Kuan Yeh*, Joon Sik Kim*

Why Interpretable Methods

- Safety -- Is this car safe to ride in?

Why Interpretable Methods

- Trust -- How can I trust you?



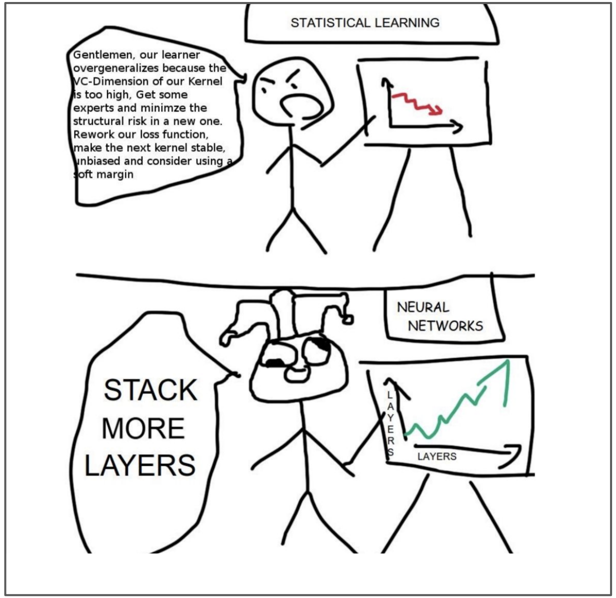
(image from Rebeiro et al.)

Why Interpretable Methods

- Learn -- How can I become a better Go player?

Why Interpretable Methods

- Improve -- How can I improve my model performance?



Interpretable Tool

These training images have multiple correct labels.

Types of Interpretable Models

Before Training

Dataset analysis

During Training

Model → Class: Dog

Model: classified as a dog since it looks like other dogs.

Interpretable Model

After Training

Model → Class: Dog

Post-hoc Explanation: the model classify this as a dog because ...

Post-hoc Explanation

Types of Interpretable Models

Local Explanations

Model

Class: Dog

Why is this image classified as a dog?

Global Explanations

Model

Class: Cat

Model

Class: Dog

Model

Class: Car

How did the model classify the images?

Types of Interpretable Models

Feature-based explanations

Model

Class: Dog

This picture is classified as a dog because of the bright pixels are used by the model:

Instance-based explanations

Model

Class: Dog

This picture is classified as a dog because of these training images are labeled as dogs:

Types of Interpretable Models

- Our model can be used in several settings:
    - Can be seen as an **interpretable model** and a **post-hoc explanation.**
    - Can be used as a **global explanation** and a **local explanation.**
    - Is mainly an **instanced-based explanation**, but can be combined with **feature-based explanations**.

Representer Theorem for RKHS

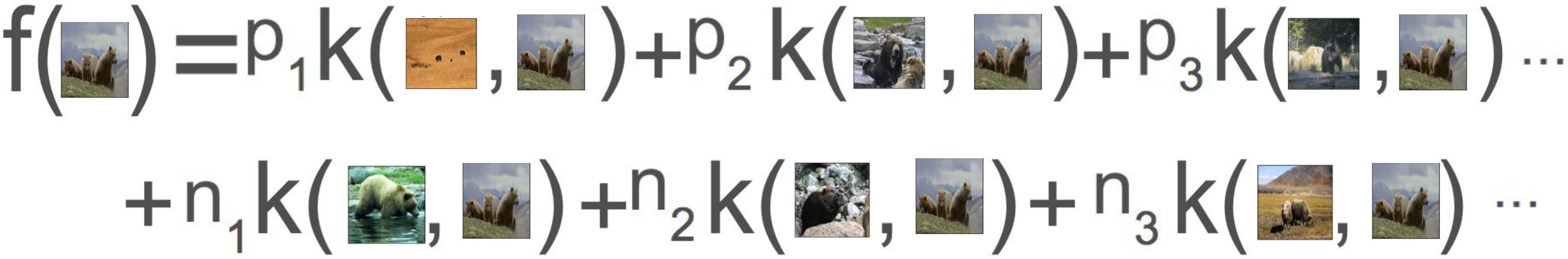
Theorem 1 (The Representer Theorem)**.** *Let k be a kernel on $\mathcal{X}$ and let $\mathcal{F}$ be its associated RKHS. Fix $x_1, \ldots, x_n \in \mathcal{X}$, and consider the optimization problem*

$$\min_{f \in \mathcal{F}} D(f(x_1), \ldots, f(x_n)) + P(\|f\|_{\mathcal{F}}^2), \tag{2}$$

where P is nondecreasing and D depends on f only though $f(x_1), \ldots, f(x_n)$. If (2) has a minimizer, then it has a minimizer of the form $f = \sum_{i=1}^{n} \alpha_i k(\cdot, x_i)$ where $\alpha_i \in \mathbb{R}$. Furthermore, if P is strictly increasing, then every solution of (2) has this form.

Representer Point Selection for Explaining Deep Neural Network

- We can show that



for some positive $p_1\ p_2\ p_3 \ldots$ and negative $n_1\ n_2\ n_3 \ldots$ and a kernel function k. This shares the form of Representer Theorem in RKHS space.

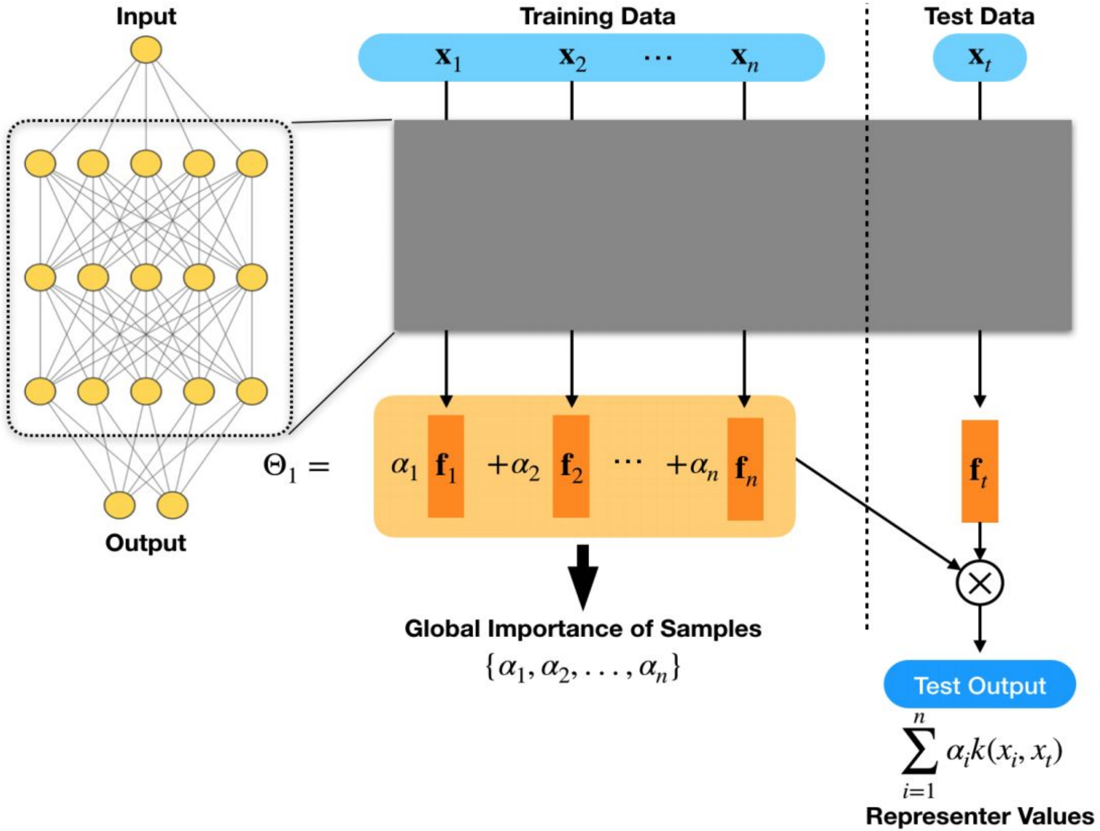
Representer Point Selection for Explaining Deep Neural Network

- We enhance the understanding of a neural network prediction by pointing to a set of representer points in the training set.

Intuition

- Most neural networks can be seen as first performing feature extraction and then performing classification.
- We can view the dot product of the features between two data point as a similarity measure (or a kernel function).
- We show that the prediction of a data point can be written as a linear combination of the similarity between the data point and training instances (under certain conditions).

Illustration



Formal Theorem Statement

Theorem 3.1. *Let us denote the neural network prediction function by* $\hat{\mathbf{y}}_i = \sigma(\Phi(\mathbf{x}_i, \boldsymbol{\Theta}))$, *where* $\Phi(\mathbf{x}_i, \boldsymbol{\Theta}) = \boldsymbol{\Theta}_1 \mathbf{f}_i$ *and* $\mathbf{f}_i = \Phi_2(\mathbf{x}_i, \boldsymbol{\Theta}_2)$. *Suppose* $\boldsymbol{\Theta}^*$ *is a stationary point of the optimization problem:* $\arg\min_{\boldsymbol{\Theta}} \{\frac{1}{n}\sum_i^n L(\mathbf{x}_i, \mathbf{y}_i, \boldsymbol{\Theta})) + g(||\boldsymbol{\Theta}_1||)\}$, *where* $g(||\boldsymbol{\Theta}_1||) = \lambda||\boldsymbol{\Theta}_1||^2$ *for some* $\lambda > 0$. *Then we have the decomposition:*

$$\Phi(\mathbf{x}_t, \boldsymbol{\Theta}^*) = \sum_i^n k(\mathbf{x}_t, \mathbf{x}_i, \alpha_i),$$

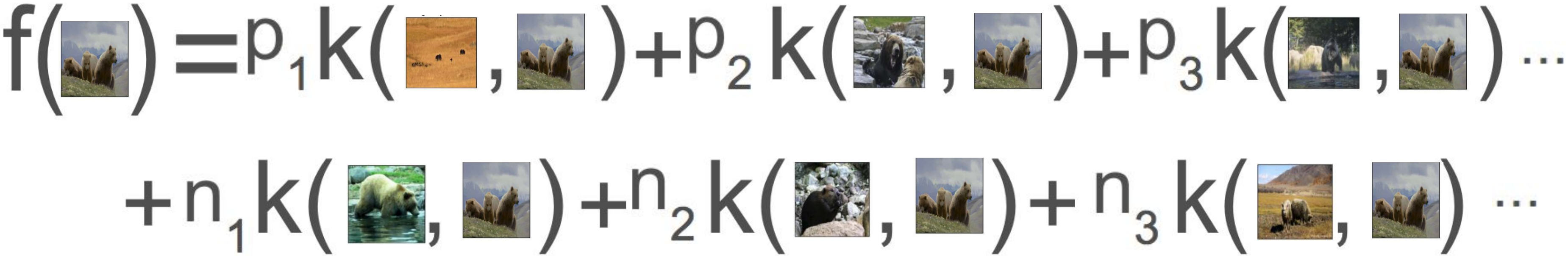
where $\alpha_i = \frac{1}{-2\lambda n} \frac{\partial L(\mathbf{x}_i, \mathbf{y}_i, \boldsymbol{\Theta})}{\partial \Phi(\mathbf{x}_i, \boldsymbol{\Theta})}$ *and* $k(\mathbf{x}_t, \mathbf{x}_i, \alpha_i) = \alpha_i \mathbf{f}_i^T \mathbf{f}_t$, *which we call a representer value for* $\mathbf{x}_i$ *given* $\mathbf{x}_t$.

Proof

- Proof is simple. By taking the gradient to be 0, the weight of last fully connected layer can be written as linear combination of training point features.
- Therefore, the prediction of the testing point is a linear combination of dot product of testing and training point features.

Theorem Interpretation

- The prediction of a testing point is determined by its similarity to positive training images and negative training images. If the feature is closer to positive training images and further away from negative training images, the prediction score will be higher and vice versa.



Some Use Cases

Training an Interpretable Model with L2 Regularization

$$\mathbf{\Theta}^* = \arg\min_{\mathbf{\Theta}} \frac{1}{n} \sum_{i}^{n} L(\mathbf{y}_i, \Phi(\mathbf{x}_i, \mathbf{\Theta})) + \lambda||\mathbf{\Theta}_1||^2$$

Some Use Cases

Post-hoc Analysis of a Given Pre-trained Model

$$\mathbf{\Theta}^* \in \arg\min_{\mathbf{\Theta}} \left\{ \frac{1}{n} \sum_{i}^{n} L(\Phi(\mathbf{x}_i, \mathbf{\Theta}_{given}), \Phi(\mathbf{x}_i, \mathbf{\Theta})) + \lambda ||\mathbf{\Theta}_1||^2 \right\}$$

for any $\mathbf{\Theta}^* \in \arg\min_{\mathbf{\Theta}} L(\Phi(\mathbf{x}_i, \mathbf{\Theta}_{given}), \Phi(\mathbf{x}_i, \mathbf{\Theta}))$,
we have $\sigma(\Phi(\mathbf{x}_i, \mathbf{\Theta}^*)) = \sigma(\Phi(\mathbf{x}_i, \mathbf{\Theta}_{given}))$.

Experiments

1. Visualizations of Positive/Negative Representer Points
2. Misclassification Analysis
3. Sensitivity Map Decomposition
4. Dataset Debugging
5. Computational Cost / Numerical Stability

Datasets: CIFAR10, Animals with Attributes (AwA)

Positive and Negative Representer Points (1)

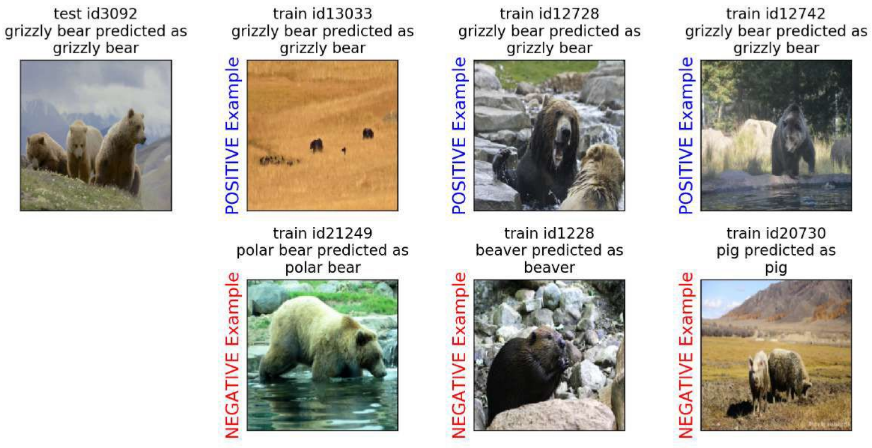
$$\Phi(\mathbf{x}_t, \Theta^*) = \sum_{i}^{n} \alpha_i \mathbf{f}_i^T \mathbf{f}_t$$

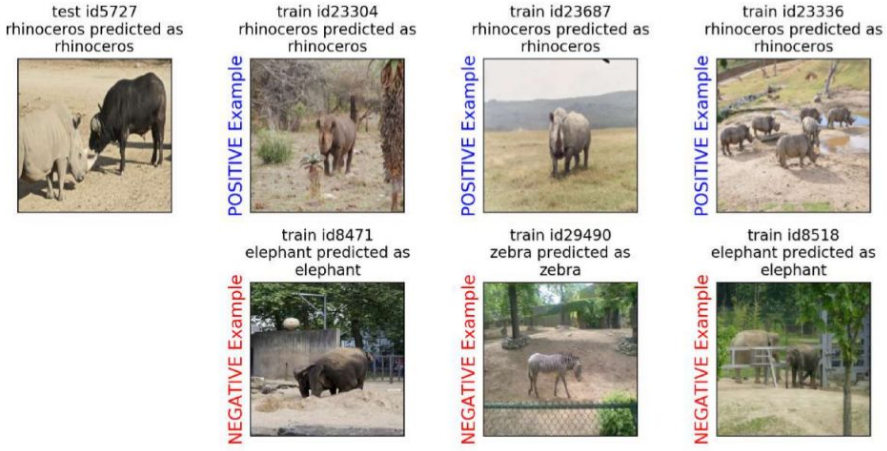
Global sample importance (α_i) — Feature similarity ($\mathbf{f}_i^T \mathbf{f}_t$)

- Positive Representer Points (Excitatory)
  - Positive global sample importance + Positive feature similarity
  - Negative global sample importance + Negative feature similarity
- Negative Representer Points (Inhibitory)
  - Negative global sample importance + Positive feature similarity
  - Positive global sample importance + Negative feature similarity

Positive and Negative Representer Points (2)

- Visualization on AwA Dataset





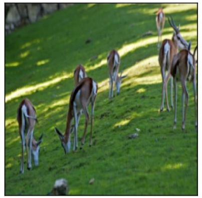
Making Sense of Misclassifications

- Can we understand why the model made a misclassification?

test id7
predicted as deer
true label is antelope

train id29372
predicted as zebra
true label is zebra

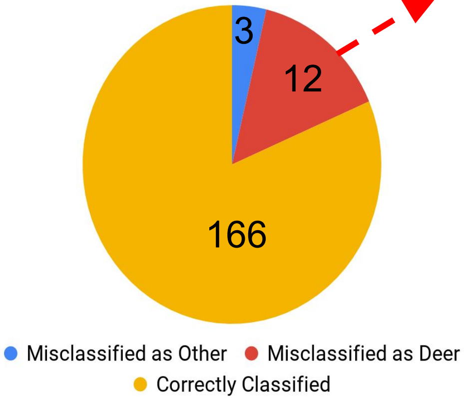
train id688
predicted as deer
true label is antelope



train id8090
predicted as elephant
true label is elephant

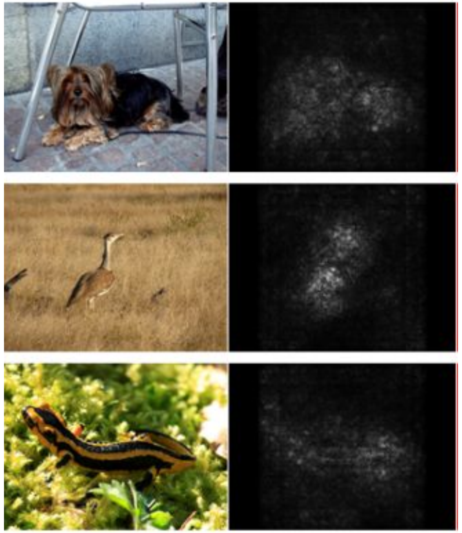
train id29208
predicted as zebra
true label is zebra

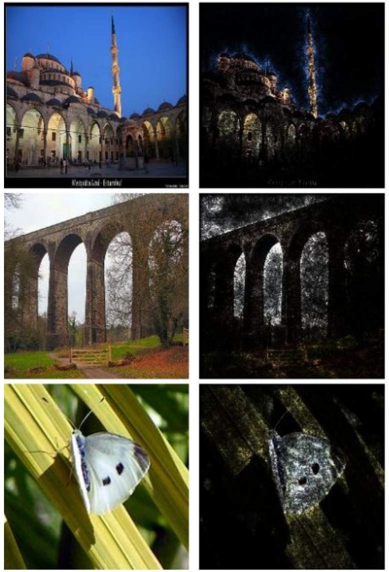
Test Points with Labels Antelope



Sensitivity Map Decomposition (1)

- Sensitivity Map: indication of how each feature influences the prediction
  - Saliency maps (Simonyan et al. 2013), LRP (Bach et al. 2015), Integrated Gradients (Sundararajan et al. 2017), SmoothGrad (Smilkov et al. 2017) etc.





Samples taken from Simonyan et al., Sundararajan et al.

Sensitivity Map Decomposition (2)

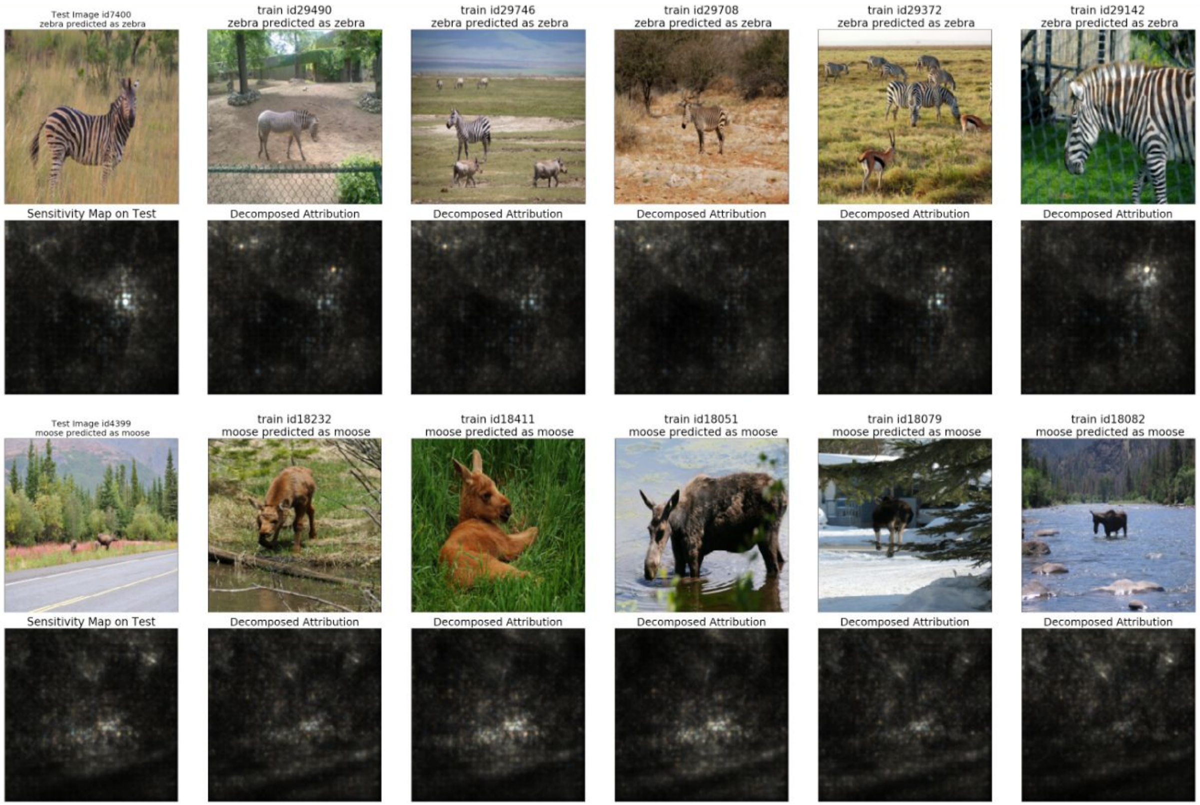
- Can we decompose sensitivity map using representer values, in terms of each training points?

$$\Phi(\mathbf{x}_t, \Theta^*) = \sum_i^n \alpha_i \mathbf{f}_i^T \mathbf{f}_t \quad \Longleftrightarrow \quad \frac{\partial \Phi(\mathbf{x}_t, \Theta^*)}{\partial \mathbf{x}_t} = \sum_i^n \alpha_i \frac{\partial \mathbf{f}_i^T \mathbf{f}_t}{\partial \mathbf{x}_t}$$

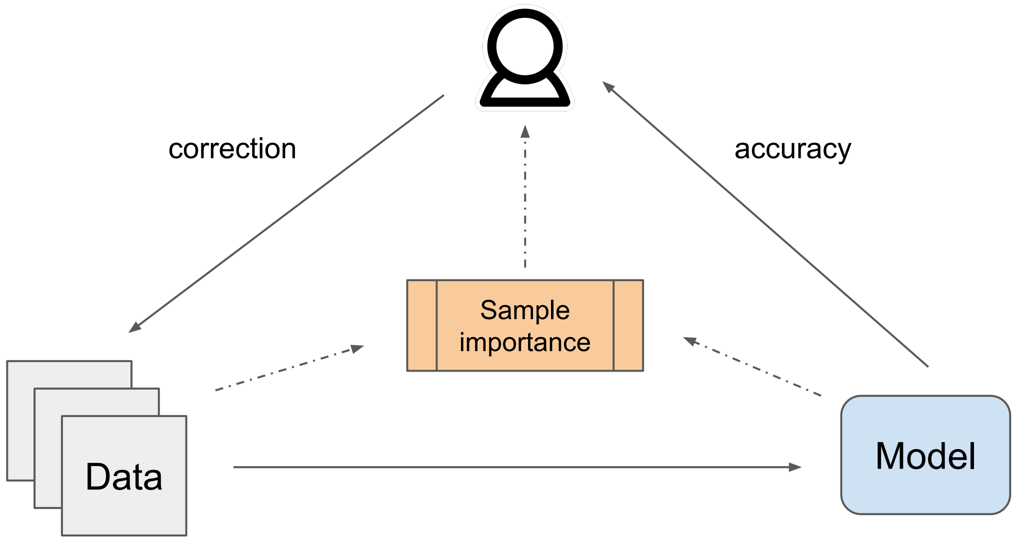
Sensitivity map

Weighted sum of sensitivity maps specific to each training points

Sensitivity Map Decomposition (3)



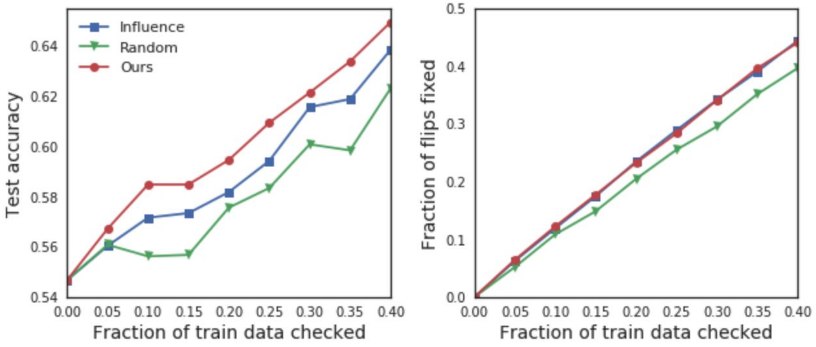
Dataset Debugging (1)



- Given a training dataset with corrupted labels, can we correct them?
- And with the corrected dataset, can we increase the test accuracy?

Dataset Debugging (2)

- Result on CIFAR10
  - Binary classification of class automobile vs horse
  - Logistic regression model
  - Select training points with higher absolute value of α_i



Computational Cost and Numerical Stability (1)

- Can the values be computed in an efficient manner?
    - Important for scaling up / real-time computation

- Are computed values numerically stable?
    - Possible issues with downstream tasks

Computational Cost and Numerical Stability (2)

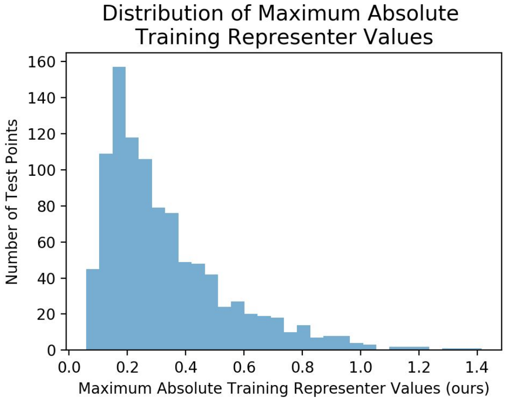
- Computational cost result on CIFAR10 and AwA dataset
  - Randomly selected 50 test points to compute influence function / representer values for all training points.

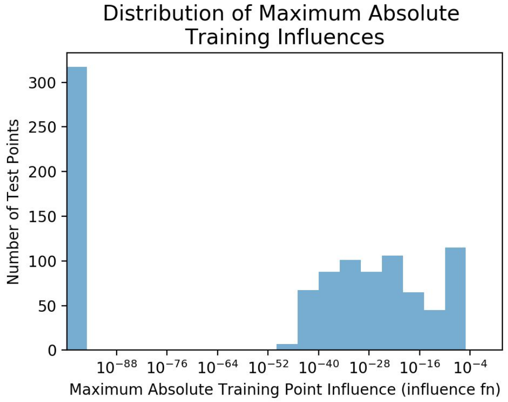
	Influence Function (Koh et al. 2017)		Representer Points (Ours)	
Dataset	Fine-Tuning	Computation	Fine-Tuning	Computation
CIFAR10	0	267.08 ± 248.20	7.09 ± 0.76	**0.10 ± 0.08**
AwA	0	172.71 ± 32.63	12.41 ± 2.37	**0.19 ± 0.12**

Measured in seconds

Computational Cost and Numerical Stability (3)

- Numerical stability result on CIFAR10 dataset
  - Randomly selected 1000 test points to compute influence function / representer values for all training points





Summary

- We prove that the deep neural network prediction of a test point can be decomposed into a linear combination of representer values of each training point.
- We illustrate the usefulness of the formulation in various use cases.
- We show that it is computationally efficient and suitable for real-time applications.

For more information ...

- Paper on Arxiv : https://arxiv.org/pdf/1811.09720.pdf
- Code on Github : https://github.com/chihkuanyeh/Representer_Point_Selection

Questions

References

Sundararajan, Mukund, Ankur Taly, and Qiqi Yan. "Axiomatic attribution for deep networks." *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017.

Simonyan, Karen, Andrea Vedaldi, and Andrew Zisserman. "Deep inside convolutional networks: Visualising image classification models and saliency maps." *arXiv preprint arXiv:1312.6034* (2013).

Bach, Sebastian, et al. "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation." *PloS one* 10.7 (2015): e0130140.

Smilkov, Daniel, et al. "Smoothgrad: removing noise by adding noise." *arXiv preprint arXiv:1706.03825* (2017).

Koh, Pang Wei, and Percy Liang. "Understanding black-box predictions via influence functions." *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017.

Appendix

Proof. Note that for any stationary point, the gradient of the loss with respect to $\mathbf{\Theta}_1$ is equal to 0. We therefore have

$$\frac{1}{n}\sum_{i=1}^{n}\frac{\partial L(\mathbf{x}_i,\mathbf{y}_i,\mathbf{\Theta})}{\partial\mathbf{\Theta}_1}+2\lambda\mathbf{\Theta}_1^*=0 \quad\Rightarrow\quad \mathbf{\Theta}_1^*=-\frac{1}{2\lambda n}\sum_{i=1}^{n}\frac{\partial L(\mathbf{x}_i,\mathbf{y}_i,\mathbf{\Theta})}{\partial\mathbf{\Theta}_1}=\sum_{i=1}^{n}\alpha_i\mathbf{f}_i^T \tag{1}$$

where $\alpha_i=-\frac{1}{2\lambda n}\frac{\partial L(\mathbf{x}_i,\mathbf{y}_i,\mathbf{\Theta})}{\partial\Phi(\mathbf{x}_i,\mathbf{\Theta})}$ by the chain rule. We thus have that

$$\Phi(\mathbf{x}_t,\mathbf{\Theta}^*)=\mathbf{\Theta}_1^*\mathbf{f}_t=\sum_{i=1}^{n}k(\mathbf{x}_t,\mathbf{x}_i,\alpha_i), \tag{2}$$

where $k(\mathbf{x}_t,\mathbf{x}_i,\alpha_i)=\alpha_i\mathbf{f}_i^T\mathbf{f}_t$ by simply plugging in the expression (1) into (2). □