On the Support Vector Effect in DNNs: Rethinking Last Layer Sensitivity-based Instance Attribution

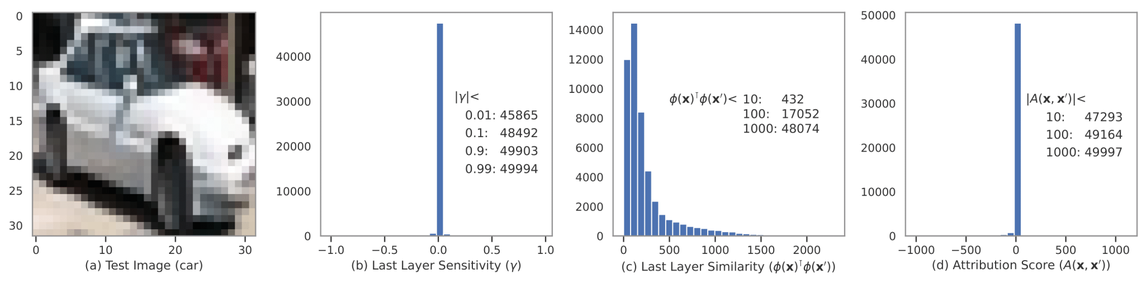
Hasan Amin, Rajiv Khanna

PURDUE UNIVERSITY
Department of Computer Science

Overview

- **Focusing on last layer in DNNs, to circumvent computational limitations, is a popular yet bad idea to find influential data points for a given prediction. Instance attribution based on simple similarity-based comparison in prediction layer is often good enough.**
- **Instance Attribution:** Selecting 'influential' training instances that the model capitalized on to make a given test prediction.
- The **Leave-One-Out (LOO)** methodology is **extremely costly**.
- Multiple methods like Influence Functions, Representer Point Selection and TracIn, which rely on loss sensitivities (for LOO approximation), proposed as **(relatively)** efficient alternatives.
- **SVE causes these methods to behave like class-level differentiators rather than instance-level explainers.** At an extreme, ***all*** *correct* predictions from a class could be attributed to a ***single*** *mislabeled* training instance.

Prediction-As-Embedding (PAE)



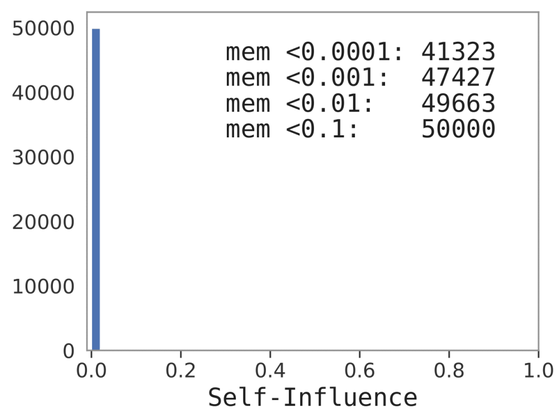
- Attribution scores by many existing methods can be seen as product of sensitivity and similarity terms. The former is problematic due to SVE, so how about we rely on similarity only? **(Back Towards) Similarity-based Instance Attribution**
- While penultimate layer similarity has been explored in the past, we propose to go a step further and use Prediction As Embedding (PAE), utilizing class conditional probabilities for similarity-based attribution:
  - **intuitive** realization of how model sees data points
  - **faithful** explanation that keeps underlying model unchanged
  - **versatile** since can work with any predictive model without needing access to model architecture, gradients or training data
  - **efficient** as it relies on low dimensional distance computation

Support Vector Effect (SVE)

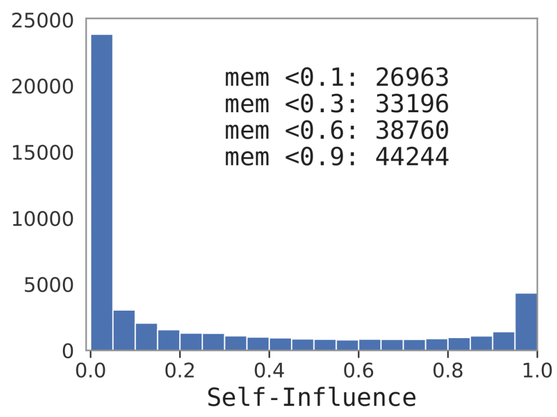
- Can view DNN training as a serial two-step process:
  - *representation learning* of meaningful last layer features
  - *linear classification* using these last layer features
- For zero training error, last layer outputs are linearly separable. Moreover, this last layer classifier exhibits behavior akin to learning maximum margin decision boundary, like a Support Vector Machine.
- **Max-margin-like behavior in the last layer(s) of DNNs.**
- **Sparse Representer Theorem (informal):** For a model trained to 0 error using SGD, we can represent prediction of any test point as a linear combination of last layer similarity with training data points, and the vector of all the alphas $\boldsymbol{\alpha}$ is sparse, i.e., $\|\boldsymbol{\alpha}\|_0 < d$.

$$\Phi(\mathbf{x}', \Theta^*) = \sum_{i=1}^{m} \alpha_i \, \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}')$$

- **When using the last layer (gradients), only a few instances—the support vectors—are considered important for a (and possibly every) prediction.**
- **Q:** Does SVE exist in practical DNNs (with higher training error, different optimizer etc.)? **Yes**, we observe memorization or empirical influence of almost all training instances to be almost 0 when using last layer representations, while reasonably spread out when using entire model with raw input and all layers.

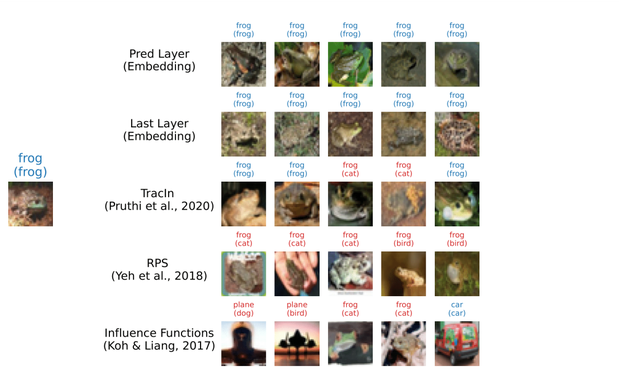


Using Last Layer

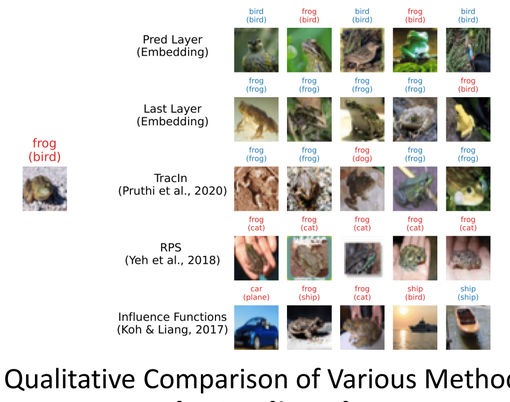


Using Input Layer

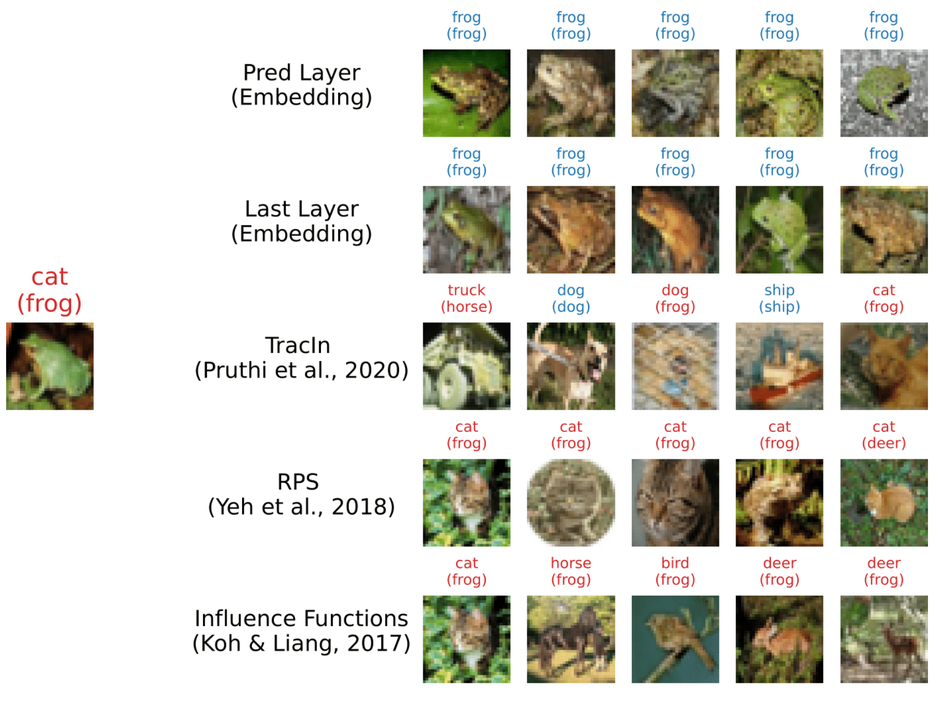
Evaluation



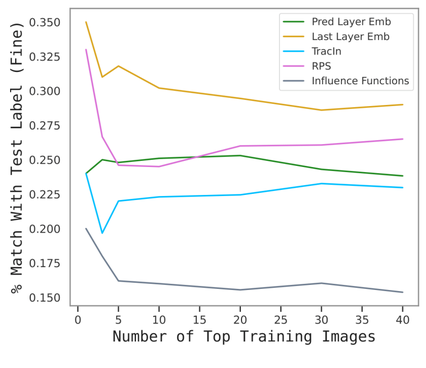
Qualitative Comparison of Various Methods on **Correctly Predicted** Test Instance



Qualitative Comparison of Various Methods on **Incorrectly Predicted** Test Instance



Qualitative Comparison of Various Methods on **Mislabeled** Test Instance



Identical Class and Subclass Tests for Various Methods on 2-class CIFAR-10B (vehicle or not). Higher is better.