Understanding and Manipulating Neural Net features Using Sparse Oblique Classification Trees

Suryabhan Singh Hada
Miguel Á. Carreira-Perpiñán   Arman Zharmagambetov
{shada,mcarreira-perpinan, azharmagambetov}@ucmerced.edu

Electrical Engineering and Computer Science,
University of California, Merced
http://eecs.ucmerced.edu

July 20, 2021

Introduction

- Deep neural nets have become the preferred model in a number of practical problems, such as computer vision, language processing, games, self-driving cars, and other engineering applications.
- The way neural nets are defined and optimized, and the sheer size and complexity of state-of-the-art deep nets, makes them very hard to understand in explanatory terms.
- Much work has focused on understanding what part of the input pattern (an image, say) is responsible for a particular class being predicted, and how the input may be manipulated to predict a different class.
- We focus instead on understanding what internal features computed by the neural net are responsible for a particular class.

- Consider a trained deep net classifier:

$$\mathbf{y} = \mathbf{f}(\mathbf{x})$$

- We can write $\mathbf{f}$ as: $\mathbf{f}(\mathbf{x}) = \mathbf{g}(\mathbf{F}(\mathbf{x}))$, where
  - $\mathbf{F}$ represents the features-extraction part $(\mathbf{z} = \mathbf{F}(\mathbf{x}) \in \mathbb{R}^F)$.
  - $\mathbf{g}$ represents the classifier part $(\mathbf{y} = \mathbf{g}(\mathbf{z}))$.
- The last layer of $\mathbf{F}$ is interesting, as it is associated with the features extracted by $\mathbf{F}$ that goes into $\mathbf{g}$.
- We want to understand the relationship between neurons in the last layer of $\mathbf{F}$ and the classes.

What we found

- Out of thousands of neurons, there is a small subset of neurons associated with a given class.
- We explore this by introducing a new feature level adversarial attack via masking specific set of neurons.
- These attacks include to make net to predict or not predict a given class.

Our approach

- We study the relationship between neurons at the last layer of $\mathbf{F}$ and the classes using sparse oblique trees.
- Overall approach:
  - Train a sparse oblique tree $y = T(\mathbf{z})$ on the training set $\{(\mathbf{F}(\mathbf{x}_n), y_n)\}_{n=1}^{N} \subset \mathbb{R}^F \times \{1, \dots, K\}$. Choose the sparsity hyperparameter $\lambda \in [0, \infty)$ such that, T mimicks $\mathbf{g}$ very good and is as sparse as possible.
  - Inspect the tree T to create masks.

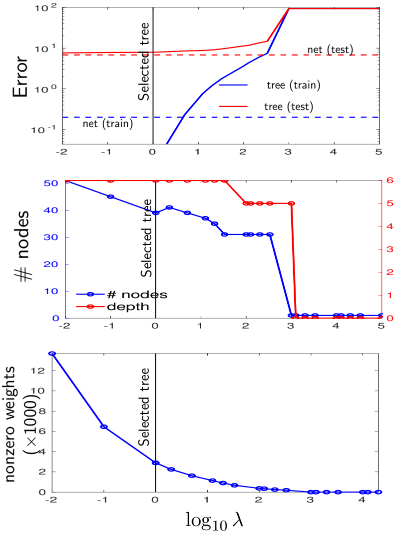
Masking of deep net features

- Objective is to control the behavior of network prediction by manipulating deep net features ($\mathbf{z} = \mathbf{F}(\mathbf{x}) \in \mathbb{R}^F$), without modifying the $\mathbf{F}$ and $\mathbf{g}$.
- Original net: $\mathbf{y} = \mathbf{f}(\mathbf{x}) = \mathbf{g}(\mathbf{F}(\mathbf{x}))$.
- Original features: $\mathbf{z} = \mathbf{F}(\mathbf{x})$.
- Masked net: $\overline{\mathbf{y}} = \overline{\mathbf{f}}(\mathbf{x}) = \mathbf{g}(\boldsymbol{\mu}(\mathbf{F}(\mathbf{x})))$
- Masked features: $\overline{\mathbf{z}} = \boldsymbol{\mu}(\mathbf{F}(\mathbf{x})) = \boldsymbol{\mu}(\mathbf{z})$.
- $\overline{\mathbf{z}} = \boldsymbol{\mu}(\mathbf{z}) = \boldsymbol{\mu}^{\times} \odot \mathbf{z} + \boldsymbol{\mu}^{+}$.
    - $\boldsymbol{\mu} = \{\boldsymbol{\mu}^{\times}, \boldsymbol{\mu}^{+}\}$
        - where, $\boldsymbol{\mu}^{\times} \in \{0,1\}^F$ is the *multiplicative mask*.
        - $\boldsymbol{\mu}^{+} \geq 0$ is the *additive mask*.

Types of masks

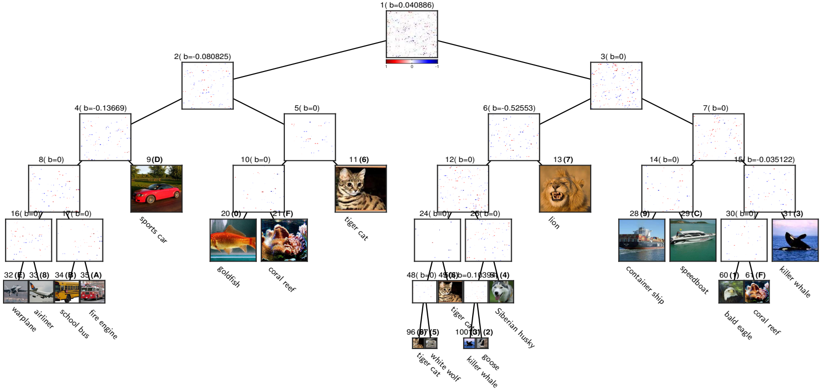
- ALL TO CLASS k.
  - Let $k \in \{1, \ldots, K\}$. Classify all instances $\mathbf{x}$ as class k.
- ALL CLASS k_1 TO CLASS k_2
  - Let $k_1 \neq k_2 \in \{1, \ldots, K\}$. For any instance originally classified as k_1, classify it as k_2. For any other instance, do not alter its classification.
- NONE TO CLASS k
  - Let $k \in \{1, \ldots, K\}$. For any instance originally classified as k, classify it as any other class. For any other instance, do not alter its classification

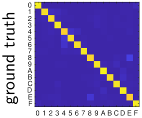
Experiments



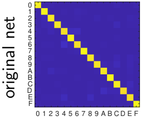
- We use VGG16 network, trained over a subset of 16 classes from ImageNet.
  - Training error: 0.2%
  - Test error: 6.79%
  - $\mathbf{z} \in \mathbb{R}^{8192}$
- We use the tree with $\lambda = 1$.
  - Training error: 0%
  - Test error: 7.9%
  - # nodes: 39
  - features used: 1366 out of 8192 (only 17%)

T with $\lambda = 1$

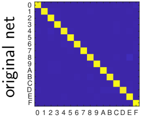




original net



tree

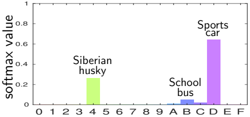


masked net

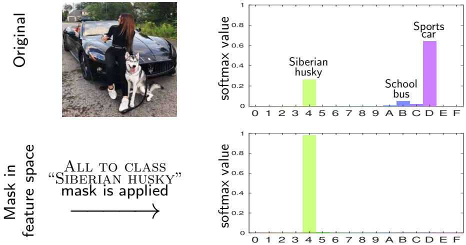
83%
neurons
masked

Mask on a single image

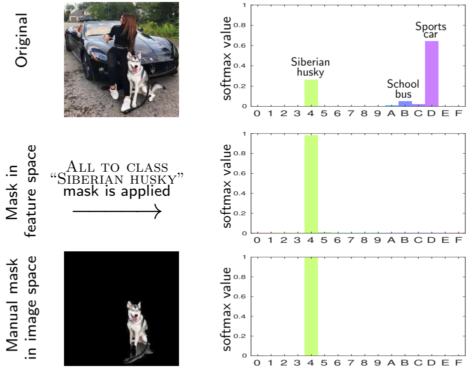
Original



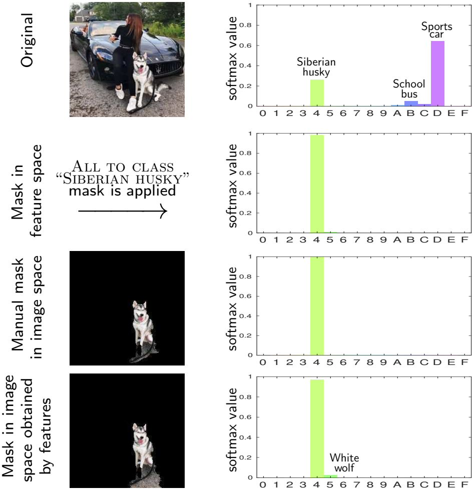
Mask on a single image



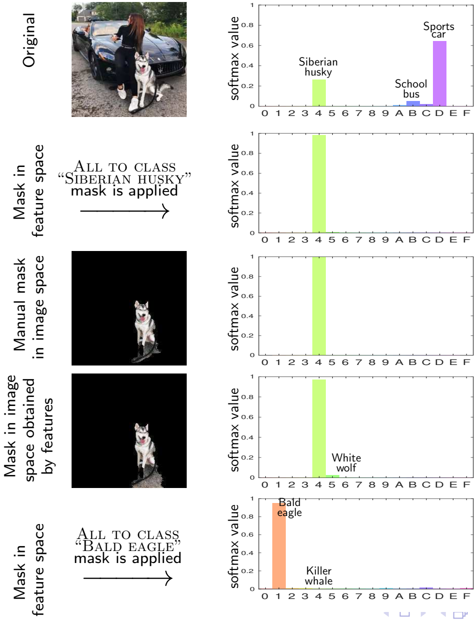
Mask on a single image

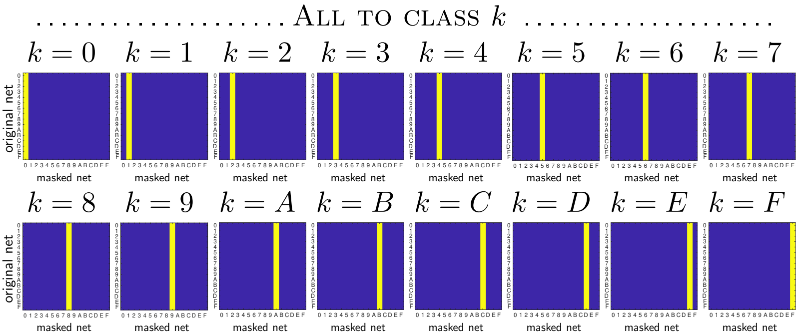


Mask on a single image

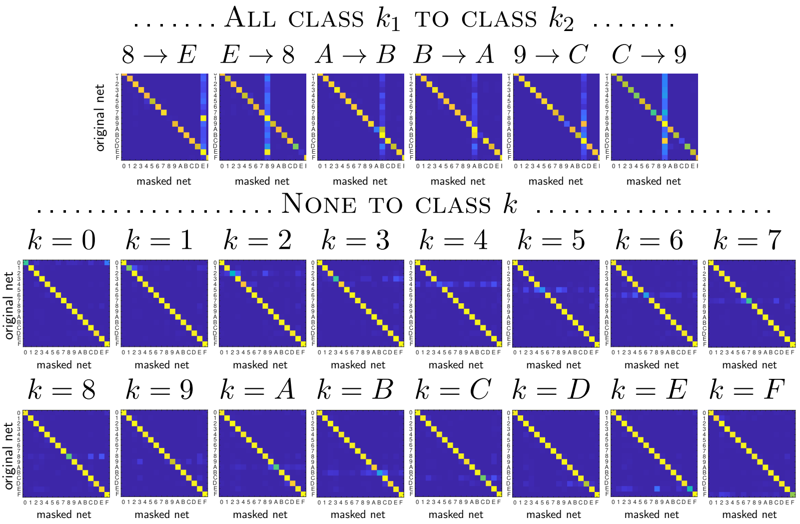


Mask on a single image



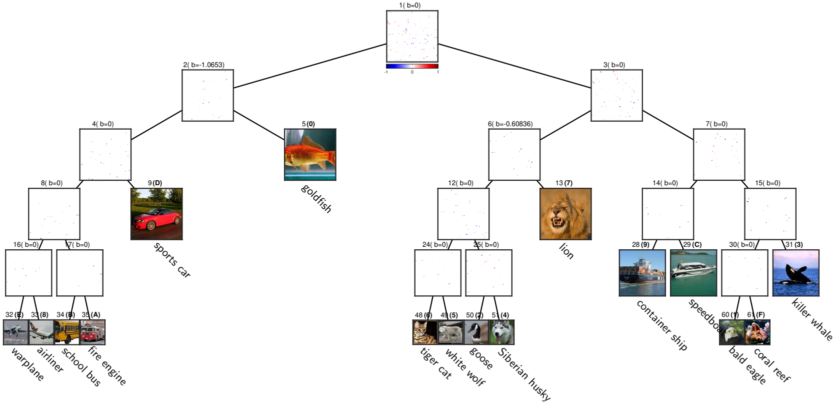

. . . . All to class k . . . .
$k = 0$
$k = 1$
$k = 2$
$k = 3$
$k = 4$
$k = 5$
$k = 6$
$k = 7$
$k = 8$
$k = 9$
$k = A$
$k = B$
$k = C$
$k = D$
$k = E$
$k = F$
original net
masked net
0 1 2 3 4 5 6 7 8 9 A B C D E F

Mask results on the test set



- Training error: 1.79%
- Test error: 9.56%

- # nodes: 31
- features used: 408 out of 8192 (only 5%)



Thank You !