How to Manipulate CNNs to Make Them Lie: the GradCAM Case

Tom Viering, Ziqi Wang, Marco Loog, Elmar Eisemann
BMVC 2019 Workshop
Interpretable & Explainable Machine Vision
What is an Explanation?

- Explain CNN decisions using heatmaps
- Blue pixels: more important for CNN decisions

Why is this image classified as monkey?
Why are explanations important?


Amazon reportedly scraps internal AI recruiting tool that was biased against women

The secret program penalized applications that contained the word “women’s”

By James Vincent | Oct 10, 2018, 7:09am EDT
How can CNNs be manipulated?

- Republishing model weights ('porting' to another framework)
- Outsourcing training to the cloud

CNN backdoor triggered by sticker [2]

Overview

• Attacks manipulate weights and architecture of already trained CNN
• CNN performance is maintained
• Explanation of GradCAM [3] is manipulated
• Lie: *explanation* is incorrect but prediction correct

CNN

Gradcam explanation

GRADCAM
CNN

featuremaps $A^1, ..., A^3$

Conv layers

$\sum \alpha_c^i A^i$

global avg pooling

Gradcam explanation

Relu

GRADCAM

backprop

TCU Delft
Attack 1

\[ A^4 = 10000 = C_A \]

For every pixel

\[ \alpha_c \]

\[ \alpha^1_c \quad \alpha^2_c \quad \alpha^3_c \quad \alpha^4_c \]

\[ \sum \alpha^i_c A^i \]

\[ \text{ReLU} \]

\[ \text{global avg pooling} \]

\[ \text{backprop} \]
Attack 2

Feature maps $A^1, ..., A^4$

Conv layers

Backprop

Global avg pooling

$\sum \alpha_c^i A^i$

ReLU
Attack 3

Random CNN

Conv layers

ReLU

\[ \sum_{i} \alpha^i_c A^i \]

\[ \alpha^1_c, \alpha^2_c, \alpha^3_c, \alpha^4_c \]

global avg pooling

backprop

desired explanation

TC2

FC1

FC2

\[ A^i \]
Attack 3

\[ G(x) = \epsilon \mod \left( v^T \tilde{A}_4, 1 \right) \]

\[ v = \tilde{1} C_v, \quad C_v \gg \epsilon, \quad \epsilon \ll 1 \]
Attack 4: backdoor

Normal image

Image with pattern

Normal explanation

Manipulated explanation
Attack 4

- Smiley detector
- Conv layers
- \[ \sum \alpha_c^i A^i \]
- ReLU
- Global avg pooling
- Backprop

Desired explanation

- Conv layers
- G
- FC1, FC2
- +

TU Delft
Attack 4

CNN + explanation seems to work fine! 😊
# Overview

<table>
<thead>
<tr>
<th></th>
<th>Attack 1 &amp; 2</th>
<th>Attack 3 &amp; 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static attack</td>
<td></td>
<td>Dynamic attack</td>
</tr>
<tr>
<td>Only extra filter and FC weights</td>
<td></td>
<td>Need extra branch, nonstandard function $G$</td>
</tr>
<tr>
<td>Architecture change reveals attack?</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Visualizing explanations reveals attack?</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
Experiment & Results

- ILSVRC 2012 (Imagenet) validation set
- VGG-16

- Accuracy changes at most of 0.002%
- Distance between observed and desired explanation on average 0.06 in $L_1$ distance
Discussion

- GradCAM is not ‘broken’, but does not *always* work!
  - Does not work if attacked
  - Other (more natural) cases where GradCAM doesn’t work?
  - Under what circumstances does it work?

- Models with similar predictions should return similar explanations?
  - Would rule out our attacks

- Future work: attack without architectural changes
  - Attack only contained in weights
  - Very hard to detect
Conclusion

• GradCAM output cannot always be trusted!
Thanks!

Tom Viering, Ziqi Wang, Marco Loog, Elmar Eisemann
The diagram illustrates an attack on a neural network model. The network consists of convolutional layers, feature maps, fully connected layers (FC1, FC2), and global average pooling. The attack is represented by the equation:

$$\tilde{h}_{new} = W_1 \tilde{A}_1 + W_2 \tilde{A}_2 + W_3 \tilde{A}_3 + W_4 \tilde{A}_4 + \tilde{b}_{new}$$

Where:
- $W_4 = C_W \mathbf{1}$, $C_W \gg 1$, $\mathbf{1}$: all-ones,
- $\tilde{b}_{new} = \tilde{b}_{old} - n_A C_A C_W$, where $n_A = \#\text{pixels} A_4$, then $h_{new} = h_{old}$. 

The goal of the attack is to manipulate the final output $h$ by adjusting the feature maps $A^i$ through backpropagation.
## Results 1-3

**Desired - Actual explanation**

**Change in score (output before softmax)**

<table>
<thead>
<tr>
<th></th>
<th>Accuracy</th>
<th>$|y_o - y_n|_\infty$</th>
<th>$|\tilde{I}_T - \tilde{I}_n|_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original network</td>
<td>0.71592</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>T1: constant</td>
<td>0.71594</td>
<td>0.01713</td>
<td>0.00513</td>
</tr>
<tr>
<td>T2: smiley</td>
<td>0.71594</td>
<td>0.00454</td>
<td>0.01079</td>
</tr>
<tr>
<td>T3: random</td>
<td>0.71592</td>
<td>0.00000</td>
<td>0.05932</td>
</tr>
</tbody>
</table>

Table 1: Evaluation of manipulated networks T1-T3 on the ILSVRC2012 validation set.
Results 4

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Network</th>
<th>Accuracy</th>
<th>$|y_o - y_n|_\infty$</th>
<th>$|\tilde{I}_T - \tilde{I}_n|_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td>Original</td>
<td>0.71592</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>T4: backdoor</td>
<td>0.71592</td>
<td>0.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>Manipulated (sticker)</td>
<td>Original</td>
<td>0.69048</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>T4: backdoor</td>
<td>0.69048</td>
<td>0.00000</td>
<td>0.00006</td>
</tr>
</tbody>
</table>

Table 2: Evaluation of Technique 4 on the ILSVRC2012 validation set.