Computational Systems Biology
Deep Learning in the Life Sciences

6.802  6.874  20.390  20.490  HST.506

Guest Lecturer: Brandon Carter

Prof. David Gifford
Lecture 5
February 20, 2020

Deep Learning Model Interpretation

http://mit6874.github.io
What’s on tap today!

• The interpretation of deep models
  – Black box methods (test model from outside)
  – White box methods (look inside of model)
  – Input dependent vs. input independent interpretations
Guess the image...
Guess the image...

traffic light
Guess the image...

traffic light
90% confidence
(InceptionResnetV2)
Why Interpretability?

- Adoption of deep learning has led to:
  - Large increase in predictive capabilities
  - Complex and poorly-understood black-box models

- Imperative that certain model decisions can be interpretably rationalized
  - Ex: loan-application screening, recidivism prediction, medical diagnoses, autonomous vehicles

- Explain model failures and improve architectures

- Interpretability is also crucial in scientific applications, where goal is to identify general underlying principles from accurate predictive models
How can we interpret deep models?
White Box Methods
(Look inside of model)

from https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/
Recall the ConvNet

AlexNet (Krizhevsky et al. 2012)

https://srdas.github.io/DLBook/ConvNets.html
Visualizing filters

Only first layer filters are interesting and interpretable

layer 1 weights

AlexNet: 64 x 3 x 11 x 11

ResNet-18: 64 x 3 x 7 x 7

ResNet-101: 64 x 3 x 7 x 7

layer 3 weights

20 x 20 x 7 x 7

from ConvNetJS CIFAR-10 demo
Visualizing activations

First layer 5th conv layer

Deconvolute node activations

Deconvolutional neural net: A novel way to map high level activities back to the input pixel space, showing what input pattern originally caused a given activation in the feature maps
Transposed convolution times received gradient is layer gradient

Convolution
3x3 filter on 4x4 input
2x2 output
Transposed convolution times received gradient is layer gradient

Convolution
3x3 filter on 4x4 input
2x2 output

Transposed Convolution
3x3 filter on 2x2 input
4x4 output
Deconvolute node activations

Layer 2

Layer 4

Layer 5

Zeiler et al., Visualizing and Understanding Convolutional Networks
Zeiler et al., Adaptive Deconvolutional Networks for Mid and High Level Feature Learning
Visualizing gradients: Saliency map

Simonyan et al., Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps
Simonyan et al., *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*
Application: Saliency maps can be used for object detection
Application: Saliency maps can be used for object detection

Simonyan et al., Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps
Application: Saliency maps can be used for object detection
Application: Saliency maps can be used for object detection

Simonyan et al., Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps
CAM: Class Activation Mapping

Use additional layer on top of the GAP (Global activation pooling) to learn **class specific** linear weights for each high level feature map and use them to weight the activations mapped back into input space.

---

Zhou et al., *Learning Deep Features for Discriminative Localization*
Use additional layer on top of the GAP (Global activation pooling) to learn **class specific** linear weights for each high level feature map and use them to weight the activations mapped back into input space.

*Zhou et al., Learning Deep Features for Discriminative Localization*
Integrated Gradients

Given an input image $x_i$ and a baseline input $x'_i$:

$$\text{IntegratedGrads}_i(x) ::= (x_i - x'_i) \times \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha(x-x'))}{\partial x_i} \ d\alpha$$

$$\text{IntegratedGrads}_i^{approx}(x) ::= (x_i - x'_i) \times \sum_{k=1}^{m} \frac{\partial F(x' + \frac{k}{m} \times (x-x'))}{\partial x_i} \times \frac{1}{m}$$

Sundararajan et al., *Axiomatic Attribution for Deep Networks*
Integrated Gradients

“Integrated Gradients”
Mukund Sundararajan, Ankur Taly, Qiqi Yan

Integrated Gradients

\[ \text{sig}(t) = \frac{1}{1 + e^{-t}} \]

Baseline:
\[ x = -8, \ y \sim 0 \]

Data point we care about:
\[ x = -8, \ y \sim 1 \]

Interesting gradients

DeepLIFT

Compares the activation of each neuron to its reference activation and assigns contribution scores according to the difference

\[
y = \max(0, x - 10)
\]

Shrikumar et al., Learning Important Features Through Propagating Activation Differences
Shrikumar et al., Not Just A Black Box: Learning Important Features Through Propagating Activation Differences
DeepLIFT

Compares the activation of each neuron to its **reference activation** and assigns contribution scores according to the difference.
Other input dependent attribution score approaches:

• LIME (Local Interpretable Model-agnostic Explanations)
  – Identify an interpretable model over the representation that is locally faithful to
    the classifier by approximating the original function with linear (interpretable) model

• SHAP (SHapley Additive explanation)
  – Unified several additive attribution score methods by using definition of Shapley
    values from game theory
  – Marginal contribution of each feature, averaged over all possible ways in which
    features can be included/excluded

• Maximum entropy
  – Locally sample inputs that maximize the entropy of predicted score
Input independent visualization: gradient ascent

Generate input that maximizes activation of certain neuron or final activation of the class

\[
\arg \max_I S_c(I) - \lambda \|I\|_2^2
\]

Simple regularizer: Penalize L2 norm of generated image

Simonyan et al., *Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps*
Input independent visualization: gradient ascent

Generate input that maximizes activation of certain neuron or final activation of the class

$$\arg \max_I S_c(I) - \lambda \|I\|_2^2$$

Simple regularizer: Penalize L2 norm of generated image

Yosinski et al., Understanding Neural Networks Through Deep Visualization
Black box methods
(Do not look inside of model)

\[ [x_1, x_2, \ldots, x_n] \xrightarrow{} F \xrightarrow{} y \]
Sufficient Input Subsets

- One simple rationale for why a black-box decision is reached is a sparse subset of the input features whose values form the basis for the decision.
- A **sufficient input subset** (SIS) is a minimal feature subset whose values alone suffice for the model to reach the same decision (even without information about the rest of the features’ values).

Carter et al., *What made you do this? Understanding black-box decisions with sufficient input subsets*
SIS help us understand misclassifications

Misclassifications

5 (6)

6

9 (9)

9

9 (4)

5

Adversarial Perturbations

5 (0)

9

9
Formal Definitions – Sufficient Input Subset

- Black-box model that maps inputs $x \in \mathcal{X}$ via a function $f : \mathcal{X} \rightarrow \mathbb{R}$
- Each input has indexable features $x = [x_1, \ldots, x_p]$ with each $x_i \in \mathbb{R}^d$
Formal Definitions – Sufficient Input Subset

- Black-box model that maps inputs $x \in \mathcal{X}$ via a function $f : \mathcal{X} \rightarrow \mathbb{R}$
- Each input has indexable features $x = [x_1, \ldots, x_p]$ with each $x_i \in \mathbb{R}^d$
- A SIS is a subset of the input features $S \subseteq [p]$ (along with their values)
- Presume decision of interest is based on $f(x) \geq \tau$ (pre-specified threshold)
- Our goal is to find a complete collection of minimal-cardinality subsets of features $S$, each satisfying $f(x_S) \geq \tau$
- $x_S =$ input where values of features outside of $S$ have been masked
SIS Algorithm

- From a particular input: we extract **SIS-collection** of disjoint feature subsets, each of which alone suffices to reach the same model decision.
- Aim to quickly identify each sufficient subset of minimal cardinality via **backward selection** (preserves interaction between features).
- Aim to identify all such subsets (under disjointness constraint).
- Mask features outside of SIS via their average value (mean-imputation).
- Compared to existing interpretability techniques, SIS is **faithful to any type of model** (sufficiency of SIS is guaranteed), and does **not** require: gradients, additional training, or an auxiliary explanation model.
Backward Selection Visualized

Pixels: 0

Score: 0.00036826866562478244

Courtesy of Zheng Dai
SIS avoids local minima by using backward selection
Example SIS for different instances of ”4”
SIS Clustered for General Insights

- Identifying the input patterns that justify a decision across many examples helps us better understand the general operating principles of a model.

- We cluster all SIS identified across a large number of examples that received the same model decision.

- Insights revealed by our SIS-clustering can be used to compare the global operating behavior of different models.
SIS Clustering Shows CNN vs. Fully Connected Network Differences (digit 4)
SIS Clustering Shows CNN vs. Fully Connected Network Differences (digit 4)
SIS Clustering Shows CNN vs. Fully Connected Network (MLP) Differences

<table>
<thead>
<tr>
<th>Cluster</th>
<th>% CNN SIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>C₁</td>
<td>100%</td>
</tr>
<tr>
<td>C₂</td>
<td>100%</td>
</tr>
<tr>
<td>C₃</td>
<td>5%</td>
</tr>
<tr>
<td>C₄</td>
<td>100%</td>
</tr>
<tr>
<td>C₅</td>
<td>100%</td>
</tr>
<tr>
<td>C₆</td>
<td>100%</td>
</tr>
<tr>
<td>C₇</td>
<td>100%</td>
</tr>
<tr>
<td>C₈</td>
<td>100%</td>
</tr>
<tr>
<td>C₉</td>
<td>0%</td>
</tr>
</tbody>
</table>

- CNN: spatially-contiguous strokes comprising small portion of digit
- MLP: decision based on pixels throughout digit, relies on global shape
- CNN is more susceptible to mistaking other (non-digit) handwritten characters for 4 if they happen to share some of the same strokes
Applying SIS to Natural Language

- We use a dataset of beer reviews from BeerAdvocate [McAuley et al. 2012]

- Different LSTM networks are trained to predict user-provided numerical ratings of aspects like *aroma*, *appearance*, and *palate*
on tap at the brewpub december 27 2010 pours a dark brown color with a **good tan** head that leaves behind a bit of **lacing** and sticks around for awhile the **nose** is really nice and **chocolatey** really love the level they 've used under that a bit of roasted malt but this was mostly about the chocolate the **taste** is n't quite as nice though the chocolate notes really still stand out the feel was quite **nice** with a **full body** pretty **viscous** for what it is drinks quite well i 'm a big fan
on tap at the pour is a dark amber color bordering on mahogany with a finger 's worth of slightly off white head s wow

the nose on this beer is phenomenal tons of vanilla bourbon maple syrup brown sugar caramel and toffee provide a

wonderful sweetness some dark fruit notes and chocolate fill in the background of the aroma t the flavor is similarly

impressive lots of sweet rich vanilla bourbon and oak accompanied by toffee caramel brown sugar and maple syrup the

finish is all that prevents this from a perfect score as there is a bit of alcohol and heat but there are some nice hints of

chocolate m the mouthfeel is smooth creamy rich and full bodied a light but nearly perfect level of carbonation d i was
told this beer was good but i had to see for myself this is one of if not the best barrel aged barleywines i 've come across i
might go back again soon to have some more
SIS Produces Minimal Sufficient Subsets

![Graph showing prediction on rationale vs. % of text in rationale. The x-axis represents the percentage of text in the rationale, while the y-axis represents the prediction on rationale only. Various markers indicate different methods: SIS, Suff. IG, Suff. LIME, Suff. Perturb, IG, LIME, and Perturb. Each method shows distinct clustering patterns.](image-url)
SIS Clustering Shows LSTM/CNN Differences

<table>
<thead>
<tr>
<th>Clu.</th>
<th>% LSTM</th>
<th>SIS #1</th>
<th>SIS #2</th>
<th>SIS #3</th>
<th>SIS #4</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>0%</td>
<td>delicious</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>C2</td>
<td>0%</td>
<td>very nice</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>C3</td>
<td>20%</td>
<td>rich chocolate</td>
<td>very rich</td>
<td>chocolate complex</td>
<td>smells rich</td>
</tr>
<tr>
<td>C4</td>
<td>33%</td>
<td>oak chocolate</td>
<td>chocolate</td>
<td>raisins</td>
<td>oak bourbon</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>raisins</td>
<td>chocolate oak</td>
<td>raisins</td>
</tr>
<tr>
<td>C5</td>
<td>70%</td>
<td>complex aroma</td>
<td>aroma complex</td>
<td>aroma complex</td>
<td>aroma complex</td>
</tr>
</tbody>
</table>
Example sufficient input subsets for MAFF binding

Two DNA sequences that receive positive TF (MAFF) binding predictions (SIS is shaded):

```
CACTGTCATTCTCTTGTCAGCCCTGGACATCCCTGGAAAGGATGACTCAGCTGTCCGTCTTTTAACAGGGTAGTTCAAGAGAATACATTCTCTGGTTATTCA
TTTTTTCTCCCTTCCACACTATGATTTCATTTCTTCTTTTGTCCTGACGTGGTTTTTCCTAAATTTCTTAGGGTGAACACTGA
```
Example clustered SIS for a transcription factor (MAFF factor)

Clustering results for a particular TF (MAFF), two clusters were found:

<table>
<thead>
<tr>
<th>SIS</th>
<th>Freq.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCTGAGTCAT</td>
<td>197</td>
</tr>
<tr>
<td>ATGACTCAGC</td>
<td>185</td>
</tr>
<tr>
<td>GCTGAGTCAC</td>
<td>83</td>
</tr>
<tr>
<td>GCTGAGTCAC</td>
<td>53</td>
</tr>
<tr>
<td>GCTGACTCAGCA</td>
<td>42</td>
</tr>
<tr>
<td>TGCTGA--GCA-TTT</td>
<td>12</td>
</tr>
<tr>
<td>GCTGAC--GCA-TTT</td>
<td>8</td>
</tr>
<tr>
<td>TGCTGAC--GCA-TT</td>
<td>6</td>
</tr>
<tr>
<td>TGCTGAC--GCA-AA</td>
<td>5</td>
</tr>
<tr>
<td>TGCTGAC--GCA-AT</td>
<td>4</td>
</tr>
</tbody>
</table>

Right image: known JASPAR motif (top) and alignment with cluster modes (bottom)
FIN - Thank You
SIS Resources

SIS paper:
https://arxiv.org/abs/1810.03805

Code for open-source SIS library and tutorial:
https://github.com/google-research/google-research/tree/master/sufficient_input_subsets