Interpreting and Explaining Deep Neural Networks: A Perspective on Time Series Data

Agenda (150 min)

Overview to Explainable Artificial Intelligence (XAI) – 15 min
- Biases in AI systems
- General Data Protection Regulation (GDPR)
- Categories of XAI algorithms

Input Attributions Methods for Deep Neural Networks – 35 min
[10 min break]

Interpreting Inside of Deep Neural Networks – 50 min
[10 min break]

Explainable Models for Time Series Data – 50 min
In 2025, estimated economic impact of ‘Automation of Knowledge work’ may reach up to 6.7 trillion US dollar. In US, 51% of US wages or $2.7 trillion in wages could be automated.
Semantic Segmentation by SegNet 2015
Pyramid Scene Parsing Network

CVPR 2017

Hengshuang Zhao¹ Jianping Shi² Xiaojuan Qi¹ Xiaogang Wang¹ Jiaya Jia¹
¹The Chinese University of Hong Kong ²SenseTime Group Limited
Many, complex AI systems are not transparent to see the mechanisms inside!

Uber’s first car accident - Death of Elaine Herzberg

Uber's self-driving car killed a pedestrian (March 18th, 2018)
The ‘safety driver’ was watching a TV show (June 22th, 2018)
## COMPAS: Prediction of Crime

<table>
<thead>
<tr>
<th>Prior Offense</th>
<th>1 attempted burglary</th>
<th>1 resisting arrest without violence</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPAS' decision</td>
<td>DYLAN FUGETT (LOW RISK 3)</td>
<td>BERNARD PARKER (HIGH RISK 10)</td>
</tr>
<tr>
<td>Subsequent Offenses</td>
<td>3 drug possessions</td>
<td>None</td>
</tr>
</tbody>
</table>

**Do We Understand AI Systems Enough?**

AI algorithms are exposed to

(1) data bias,
(2) model bias, and
(3) algorithmic bias
<table>
<thead>
<tr>
<th>Article</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>13-14. Right to explanation</td>
<td>A data subject has the right to “meaningful information about the logic involved” when decision is made automatically.</td>
</tr>
<tr>
<td>EU administration</td>
<td>When violated 4% of global revenue will be fined.</td>
</tr>
<tr>
<td>Enact</td>
<td>May 28th, 2018</td>
</tr>
</tbody>
</table>
Statistically impressive, but individually unreliable

Inherent flaws can be exploited

Skewed training data creates Maladaptation

A DARPA Perspective on AI – Three Waves of AI
Explainable AI – Performance vs. Explainability
Explainable AI – Performance vs. Explainability

New Approach

Create a suite of machine learning techniques that produce more explainable models, while maintaining a high level of learning performance.

Learning Techniques (today)

- Neural Nets
- Graphical Models
- Bayesian Belief Nets
- SRL
- CRFs
- HBNs
- MLNs
- Markov Models
- Ensemble Methods
- Random Forests
- Decision Trees

Deep Learning

Deep Explanation

Modified deep learning techniques to learn explainable features

Explainability (notional)

- Prediction Accuracy
- Explainability

DARPA
New Approach

Create a suite of machine learning techniques that produce more explainable models, while maintaining a high level of learning performance.

Learning Techniques (today)

- Neural Nets
- Deep Learning
- Graphical Models
- Bayesian Belief Nets
- Ensemble Methods
- SRL
- CRFs
- HBNs
- Random Forests
- MLNs
- Markov Models
- Decision Trees

Explainability (notional)

Prediction Accuracy vs. Explainability

Deep Explanation
Modified deep learning techniques to learn explainable features

Interpretable Models
Techniques to learn more structured, interpretable, causal models

Explainable AI – Performance vs. Explainability
New Approach

Create a suite of machine learning techniques that produce more explainable models, while maintaining a high level of learning performance.

Explainable AI – Performance vs. Explainability
Explainable AI – Performance vs. Explainability

New Approach

Create a suite of machine learning techniques that produce more explainable models, while maintaining a high level of learning performance.

Learning Techniques (today)

- Neural Nets
- Graphical Models
- Ensemble Methods
- Bayesian Belief Nets
- SRL
- HBN
- Random Forests
- CRFs
- MLNs
- Markov Models
- Decision Trees

Explainability (notional)

Prediction Accuracy vs. Explainability

Deep Explanation

Modified deep learning techniques to learn explainable features

Interpretable Models

Techniques to learn more structured, interpretable, causal models

Model Induction

Techniques to infer an explainable model from any model as a black box

DARPA
A roadmap of Explainable Artificial Intelligence
Interpreting and Explaining Deep Neural Networks: A Perspective on Time Series Data

Agenda (150 min)

Overview to Explainable Artificial Intelligence (XAI) – 15 min

Input Attributions Methods for Deep Neural Networks – 35 min
- Properties of Good Attribution Methods
- Relevance Score Based Methods: Layer-wise Relevance Propagation (LRP), Gradient Based Methods: DeepLIFT,
- Equivalence of LRP and DeepLIFT
- Handling Negative Relevance Scores
- Relative Attributing Propagation

Interpreting Inside of Deep Neural Networks – 50 min

Explainable Models for Time Series Data – 50 min
An Example: General Setting for Attribution Methods
Model

- Input: N-dimensional one \( x = [x_1, \ldots, x_N] \in \mathbb{R}^N \)
- Output: C-dimensional one \( S(x) = [S_1(x), \ldots, S_C(x)] \in \mathbb{R}^C \)
- An attribution value (or relevance/contribution) of each input feature for a class \( c \)

\[
R^c = [R^c_1, \ldots, R^c_N] \in \mathbb{R}^N
\]

Definition: Input Attribution Toward an Output
Linear Regression

\[ y = w_0 + w_1 x_1 + \ldots + w_N x_N + \epsilon \]

**Example**

- \( y_c \): the future capital asset
- \( x_1 \) and \( x_2 \): two investments

\[ \mathbb{E}[y_c | x_1, x_2] = 1.05 x_1 + 1.50 x_2 \]

- The influence of the independent variables of the target

\[ R_1(x) = 1.05 \quad R_2(x) = 1.50 \]

- In fact, the attribution is the model gradient:

\[ R_i(x) = \frac{\partial y_c}{\partial x_i}(x) \]
• The influence of the independent variables of the target

\[ E[y_c|x_1, x_2] = 1.05x_1 + 1.50x_2. \]

• However, when there are two different inputs:

\[
\begin{align*}
x_1 &= \$100,000, & x_2 &= \$10,000 \\
y_c &= 1.05 \times \$100,000 + 1.50 \times \$10,000 \\
&= \$105,000 + \$15,000
\end{align*}
\]

\[ R_1(x) = 105'000 \quad R_2(x) = 15'000 \]

• We can compute the attributions as the gradient multiplied element-wise by the input:

\[ R_i(x) = x_i \cdot \frac{\partial y_c}{\partial x_i}(x) \]
Explanation Continuity

- An attribution method satisfies explanation continuity if:
  - Given a continuous prediction function $S_c(x)$, it produces continuous attributions $R^c(x)$.
  - That is, for two nearly identical data points, the model responses are nearly identical, then its explanations are.
Implementation Invariance

- \( m_1 \) and \( m_2 \): two functionally equivalent models
- For any \( x \), the models produce the same output

\[
\forall x : S_{m_1}(x) = S_{m_2}(x)
\]

- An attribution method is implementation invariant if it always produces identical attributions for \( m_1 \) and \( m_2 \).

\[
\forall (m_1, m_2, x, c) : R_{c,m_1}(x) = R_{c,m_2}(x)
\]
**Sensitivity-n**

- An attribution method satisfies sensitivity-n when the sum of the attributions for any subset of n features is equal to the variation of the output $S_c$ caused by removing the features.
- When n features are selected $x_S = [x_1,...,x_n] \subseteq x$

$$\sum_{i=1}^{n} R^c_i(x) = S_c(x) - S_c(x \setminus x_S)$$

- When n = N, this property is the efficiency property in the Shapley value.

$$\sum_{i=0}^{N} R^c_i(x) = S_c(x) - S_c(\bar{x})$$

- That is,

$$\forall x, c : \sum_{i=1}^{N} R^c_i(x) = S_c(x)$$

**Properties for Good Attribution Methods:** **Sensitivity-n**
Attribution methods in a linear model

\[ R_i^c(x) = \frac{\partial S_c(x)}{\partial x_i} \]

\[ R_i^c(x) = x_i \cdot \frac{\partial S_c(x)}{\partial x_i} \]

Sensitivity analysis

- Compute the absolute value of the partial derivative

\[ R_i^c(x) = \left| \frac{\partial S_c(x)}{\partial x_i} \right| \]

Gradient * Input

- Multiply the partial derivatives feature-wise by the input

\[ R_i^c(x) = \frac{\partial S_c(x)}{\partial x_i} \cdot x_i \]

\[ R_i = \left. \frac{\partial f}{\partial x_i} \right|_x \cdot x_i \]

Attribution Methods for Non-Linear Models
Goal of Input Attribution Methods
Definition of $\epsilon$-LRP

- $r_i^{(l)}$: relevance of unit $i$ of layer $l$
- The relevance of the target neuron $c$ is the activation of the neuron
- $z_{ij}$: the weighted activation of a neuron $i$ onto neuron $j$

- $b_j$: the additive $z_{ij} = x_i^{(l)} w_{ij}^{(l,l+1)}$

\[
  r_i^{(L)} = \begin{cases} 
    S_i(x) & \text{if unit } i \text{ is the target unit of interest} \\
    0 & \text{otherwise}
  \end{cases}
\]

\[
  r_i^{(l)} = \sum_j \frac{z_{ij}}{\sum_i' z_{i'j} + b_j + \epsilon \cdot \text{sign}(\sum_i' z_{i'j} + b_j)} r_j^{(l+1)}
\]

- In the input layer, the final attributions are $R_i^c(x) = r_i^{(1)}$
An Example of LRP

Forward propagation

\[ a^{(l+1)}_j = \sigma \left( \sum_i a^{(l)}_i w_{ij} + b^{(l+1)}_j \right) \]

Layer-wise relevance propagation

\[ R^{(l)}_i = \sum_j \frac{z^{(l)}_{i,j}}{\sum_{i'} z^{(l+1)}_{i',j}} R^{(l+1)}_j \]

[Image courtesy of Klaus Muller]
• The chain rule along a single path is the produce of the partial derivatives of all linear and nonlinear transformations along the path.

• For two units i and j in subsequent layers

\[
\frac{\partial x_j}{\partial x_i} = w_{ji} \cdot f'(z_j)
\]

• \(P_{ic}\): a set of paths connect units i and c

\[
\frac{\partial g x_c}{\partial x_i} = \sum_{p \in P_{ic}} \left( \prod w_p \prod g(z)_p \right)
\]

• When \(g() = f'(())\)

• This does work for fully-connected, convolutional, recurrent layers without multiplicative units, pooling operations

Some Notes on LRP
Proposition 1: $\epsilon - LRP$ is equivalent to the feature-wise product of the input and the modified partial derivative $\partial^g S_c(x)/\partial x_i$, with $g = g^{LRP} = f_i(z_i)/z_i$, i.e. the ratio between the output and the input at each nonlinearity.

- In ReLU or Tanh activations, $g^{LRP}(z)$ is the average gradient of the nonlinearity in $[0, z]$.

$$g^{LRP}(z) = (f(z) - 0)/(z - 0)$$
• Proof by induction. By definition, the $\epsilon$-LRP relevance of the target neuron $c$ on the top layer $L$ to be equal to the output of the neuron, $S_c$:

$$r_c^{(L)} = S_c(x) = f \left( \sum_j w_{cj}^{(L,L-1)} x_j^{(L-1)} + b_c \right)$$
• The relevance of the parent layer is:

\[
\begin{align*}
    r_{j}^{(L-1)} &= r_{c}^{L} \frac{w_{cj}^{(L,L-1)} x_{j}^{(L-1)}}{\sum_{j'} w_{c,j'}^{(L,L-1)} x_{j'}^{(L-1)} + b_{c}} \\
    &= f \left( \sum_{j'} w_{c,j'}^{(L,L-1)} x_{j'}^{(L-1)} + b_{c} \right) \frac{w_{cj}^{(L,L-1)} x_{j}^{(L-1)}}{\sum_{j'} w_{c,j'}^{(L,L-1)} x_{j'}^{(L-1)} + b_{c}} \\
    &= g^{LRP} \left( \sum_{j'} w_{c,j'}^{(L,L-1)} x_{j'}^{(L-1)} + b_{c} \right) \frac{w_{cj}^{(L,L-1)} x_{j}^{(L-1)}}{\sum_{j'} w_{c,j'}^{(L,L-1)} x_{j'}^{(L-1)} + b_{c}} \\
    &= \frac{\partial g^{LRP}}{\partial S_{c}(x)} \frac{S_{c}(x)}{x_{j}^{(L-1)}} \frac{\partial g^{LRP}}{\partial x_{j}^{(L-1)}} \frac{\partial x_{j}^{(L-1)}}{\partial x_{i}} \\
    &= \sum_{p \in P_{c}} \left( \prod_{w_{p}} \prod_{g(z)} g(z) \right)
\end{align*}
\]

Correctness of LRP: Proof continued
• For the inductive step from the hypothesis that on a layer $l$ the LRP explanation is:

$$r_i^{(l)} = \frac{\partial g_{LRP}^L}{\partial x_i^{(l)}} S_c(x) x_i^{(l)}$$

Then for layer $l-1$ it holds:

$$x_i^{(l)} = f(\sum_{j'} w_{ij'}^{(l-1)} x_{j'}^{(l-1)} + b_i)$$

• Then for layer $l-1$ it holds:

$$r_j^{(l-1)} = \sum_i r_i^{(l)} \frac{w_{ij}^{(l-1)} x_i^{(l-1)}}{\sum_j' w_{ij'}^{(l-1)} x_{j'}^{(l-1)} + b_i}$$

LRP propagation rule

$$= \sum_i \frac{\partial g_{LRP}^L}{\partial x_i^{(l)}} S_c(x) \frac{x_i^{(l)}}{\sum_j' w_{ij'}^{(l-1)} x_{j'}^{(l-1)} + b_i} x_j^{(l-1)}$$

By definition of $\frac{\partial g_{LRP}^L}{\partial x_i^{(l-1)}} x_j^{(l-1)}$
DeepLIFT Rescale

- \( \bar{x} \): baseline input

\[
\begin{align*}
    r_i^{(L)} &= \begin{cases} 
        S_i(x) - S_i(\bar{x}) & \text{if unit } i \text{ is the target unit of interest} \\
        0 & \text{otherwise}
    \end{cases} \\
    r_i^{(l)} &= \frac{\sum_j \frac{z_{ij} - \bar{z}_{ij}}{z_{ij} - \bar{z}_{ij}} r_j^{(l+1)}}{\sum_i z_{i'j} - \sum_i \bar{z}_{i'j}}
\end{align*}
\]
Proposition: $\epsilon$ – LRP is equivalent to

(i) Gradient * Input if only ReLUs are used as nonlinearities:

(ii) DeepLIFT (computed with a zero baseline) if applied to a network with no additive biases and with nonlinearities $f$ such that $f(0)=0$ (e.g., RELU or Tanh).
Integrated Gradients

- LRP and DeepLIFT replace each instant gradient by an average gradient at each nonlinearity does not necessarily result in the average gradient of the function as a whole.
- Thus the attribution method fails to satisfy implementation invariance.
- It computes attributions multiplying the input variable element-wise with the average partial derivative as the input varies from a baseline $\bar{x}$ to its final value $x$.

\[
R^c_i(x) = x_i \cdot \int_{\alpha=0}^{1} \frac{\partial S_c(\tilde{x})}{\partial (\tilde{x}_i)} \bigg|_{\tilde{x}=\bar{x}+\alpha(x-\bar{x})} d\alpha
\]

- It satisfies sensitivity-N.
<table>
<thead>
<tr>
<th>Method</th>
<th>Attribution $R^c_i(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensitivity analysis</td>
<td>$\frac{\partial S_c(x)}{\partial x_i}$</td>
</tr>
<tr>
<td>Gradient * Input</td>
<td>$x_i \cdot \frac{\partial S_c(x)}{\partial x_i}$</td>
</tr>
<tr>
<td>$\epsilon$-LRP</td>
<td>$x_i \cdot \frac{\partial g \cdot S_c(x)}{\partial x_i}$, $g = \frac{f(z)}{z}$</td>
</tr>
<tr>
<td>DeepLIFT (Rescale)</td>
<td>$(x_i - \bar{x}_i) \cdot \frac{\partial g \cdot S_c(x)}{\partial x_i}$, $g = \frac{f(z) - f(\bar{z})}{z - \bar{z}}$</td>
</tr>
<tr>
<td>Integrated Gradients</td>
<td>$(x_i - \bar{x}<em>i) \cdot \int</em>{\alpha=0}^{1} \left</td>
</tr>
</tbody>
</table>
Comparisons of Attribution Methods

[Image courtesy of Ancona Marco]
Results with Perturbation Methods
<table>
<thead>
<tr>
<th><strong>Saliency Maps</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Simonyan et al. 2015</td>
</tr>
<tr>
<td><strong>Integrated Gradients</strong></td>
</tr>
<tr>
<td>Sundararajan et al. 2017</td>
</tr>
<tr>
<td><strong>DeepLIFT</strong></td>
</tr>
<tr>
<td>Shrikumar et al. 2017</td>
</tr>
<tr>
<td><strong>LIME</strong></td>
</tr>
<tr>
<td>Ribeiro et al. 2016</td>
</tr>
<tr>
<td><strong>Gradient * Input</strong></td>
</tr>
<tr>
<td>Shrikumar et al. 2016</td>
</tr>
<tr>
<td><strong>Layer-wise Relevance Propagation (LRP)</strong></td>
</tr>
<tr>
<td>Bach et al. 2015</td>
</tr>
<tr>
<td><strong>Guided Backpropagation</strong></td>
</tr>
<tr>
<td>Springenberg et al. 2014</td>
</tr>
<tr>
<td><strong>Grad-CAM</strong></td>
</tr>
<tr>
<td>Selvaraju et al. 2016</td>
</tr>
<tr>
<td><strong>Simple occlusion</strong></td>
</tr>
<tr>
<td>Zeiler et al. 2014</td>
</tr>
<tr>
<td><strong>Meaningful Perturbation</strong></td>
</tr>
<tr>
<td>Fong et al. 2017</td>
</tr>
<tr>
<td><strong>Prediction Difference Analysis</strong></td>
</tr>
<tr>
<td>Zintgraf et al. 2017</td>
</tr>
<tr>
<td><strong>KernelSHAP/DeepSHAP</strong></td>
</tr>
<tr>
<td>Lundberg et al., 2017</td>
</tr>
</tbody>
</table>

Slides courtesy of [Marco Ancona, et. al., Explaining Deep Neural Networks with a Polynomial Time Algorithm for Shapley Values Approximation, ICML 2019]

[Image courtesy of Ancona Marco]

Some References
Issues with Positive/Negative Relevance Propagation
Handling Negative Relevance Scores During the Propagation

Do not propagate negative activations

Forward ReLU (activated)  Forward ReLU (deactivated)  Backward ReLU (activated)  Backward ReLU (deactivated)  Linear Neuron

[Image courtesy of Klaus Muller]
Relative Attributing Propagation

Woo-Jeong Nam, et. al., "Relative Attributing Propagation: Interpreting the Comparative Contributions of Individual Units in Deep Neural Networks", AAAI, 2020

\[ \psi_i = \begin{cases} \frac{\sum_j R_{i,j}^{(t)} + 1}{\sum_l R_{i,l}^{(t)}}, & m_i \text{ is activated} \\ 0, & \text{otherwise} \end{cases} \]

\[ R_{i,j}^{(t)} = R_{i,j}^{(t-1)} - \psi_i \]

First Propagation

Absolute Influence Normalization

\[ R_{i}^{(p)} = \left( \sum_l \frac{z_{i,l}^{(t)}}{z_{i,l}^{(t+1)}} \right) \cdot \left( \sum_l \frac{z_{i,l}^{(t)}}{z_{i,l}^{(t+1)}} \right) \cdot R_{i}^{(q)} \]

\[ R_{i}^{(p)} = \left( \sum_l \frac{z_{i,l}^{(t)} + z_{i,l}^{(t)}}{z_{i,l}^{(t)}} \right) \cdot \left( \sum_l \frac{z_{i,l}^{(t)}}{z_{i,l}^{(t)} + z_{i,l}^{(t)}} \right) \cdot R_{i}^{(q)} \]
Relative Attributing Propagation: Quantitative Evaluations
### Relative Attributing Propagation: Quantitative Evaluations

<table>
<thead>
<tr>
<th>Outside-Inside Ratio</th>
<th>RAP</th>
<th>LRP_{α_1β_0}</th>
<th>LRP_{α_2β_1}</th>
<th>Gradient</th>
<th>Input Gradient</th>
<th>Integrated Gradients</th>
<th>Pattern Attribution</th>
<th>Guided Backprop</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG-16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALL POS</td>
<td>0.252</td>
<td>0.616</td>
<td>-</td>
<td>0.619</td>
<td>0.989</td>
<td>1.230</td>
<td>-</td>
<td>1.069</td>
</tr>
<tr>
<td>POS</td>
<td>0.341</td>
<td>0.524</td>
<td>0.474</td>
<td>0.524</td>
<td>0.691</td>
<td>0.827</td>
<td>-</td>
<td>0.427</td>
</tr>
<tr>
<td>Res-50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALL POS</td>
<td>0.164</td>
<td>0.302</td>
<td>-</td>
<td>0.597</td>
<td>0.996</td>
<td>1.195</td>
<td>-</td>
<td>1.035</td>
</tr>
<tr>
<td>POS</td>
<td>0.166</td>
<td>0.299</td>
<td>0.429</td>
<td>0.597</td>
<td>0.689</td>
<td>0.698</td>
<td>-</td>
<td>0.296</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Segmentation Mask</th>
<th>RAP</th>
<th>LRP_{α_1β_0}</th>
<th>LRP_{α_2β_1}</th>
<th>Gradient</th>
<th>Input Gradient</th>
<th>Integrated Gradients</th>
<th>Pattern Attribution</th>
<th>Guided Backprop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Imagenet</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIX ACC</td>
<td>79.23</td>
<td>75.40</td>
<td>72.95</td>
<td>70.01</td>
<td>66.38</td>
<td>66.52</td>
<td>76.84</td>
<td>71.98</td>
</tr>
<tr>
<td>mIOU</td>
<td>62.23</td>
<td>55.78</td>
<td>50.86</td>
<td>49.30</td>
<td>44.01</td>
<td>45.90</td>
<td>58.05</td>
<td>49.87</td>
</tr>
<tr>
<td>Pascal VOC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIX ACC</td>
<td>73.91</td>
<td>70.86</td>
<td>69.43</td>
<td>68.14</td>
<td>50.01</td>
<td>52.38</td>
<td>-</td>
<td>66.92</td>
</tr>
<tr>
<td>mIOU</td>
<td>55.60</td>
<td>49.82</td>
<td>46.85</td>
<td>46.07</td>
<td>31.69</td>
<td>34.39</td>
<td>-</td>
<td>43.63</td>
</tr>
</tbody>
</table>

When perturbating pixels with negative attributions...

---

**Quantitative Performance**

---

**Relative Attributing Propagation: Quantitative Evaluations**
- Input attribution methods can compute the contributions of individual inputs.

- Under some assumptions, results of different input attribution methods are equivalent.

- Handling negative attributions are also important.
References


