Tools for interpreting how and what neural networks learn, and their applications for climate and weather

Imme Ebert-Uphoff[1,2], Ben Toms[3] and Elizabeth A. Barnes[3]
Colorado State University

1 Cooperative Institute for Research in the Atmosphere (CIRA)

2 Department of Electrical and Computer Engineering

3 Department of Atmospheric Science

NOAA Star seminar
October 28, 2019

Who we are.

Dr. Imme Ebert-Uphoff
Machine Learning Lead at CIRA & Research Faculty in the Dept. of Electrical & Computer Engineering at CSU

Topics: Machine Learning and causality theory applied to climate and weather; visualization methods for ANNs; physics-guided machine learning; ML for parameterization (new).

Ben Toms
PhD student in the Barnes Group at CSU; DOE Computational Science Graduate Fellow

Topics: subseasonal-to-seasonal and decadal prediction, machine learning and machine learning interpretability, high-performance computing

Prof. Elizabeth Barnes
Associate Professor of Atmospheric Science at CSU

Topics: large-scale atmospheric dynamics, subseasonal-to-seasonal and decadal prediction, climate change, data analysis, ML and ML visualization for climate research

Introduction

Using ML is getting easy:

- Availability of increasing computational power & of cloud resources. Can do a lot now *without* dedicated HPC resources.
- Highly efficient software packages. Easy to use.
- **Setting up + running experiments with ML methods no longer requires sophisticated computer science knowledge.**

Data:

- Mountains of data available (observations & model outputs).
- Lots of earth applications can benefit from power of machine learning.

Thus ML has arrived in the earth sciences with full force.

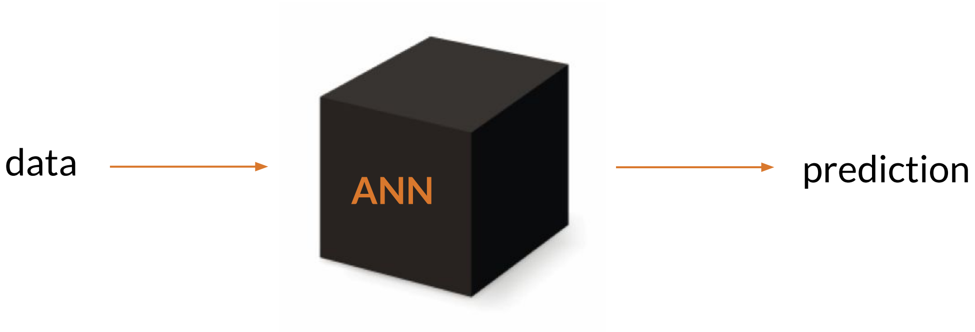
Most promising tool: **artificial neural networks (ANNs)**.

Overview of this Presentation

1. **Brief Introduction to Artificial Neural Networks**
2. **Why do we care about the strategies that ANNs use?**
3. **Background for ANN visualization tools**
4. **Layer-wise Relevance Propagation**
5. **LRP for debugging and designing ANNs**
6. **LRP for scientific discovery**
7. **Concluding Thoughts**

Brief introduction to **A**rtificial **N**eural **N**etworks

Neural networks 101

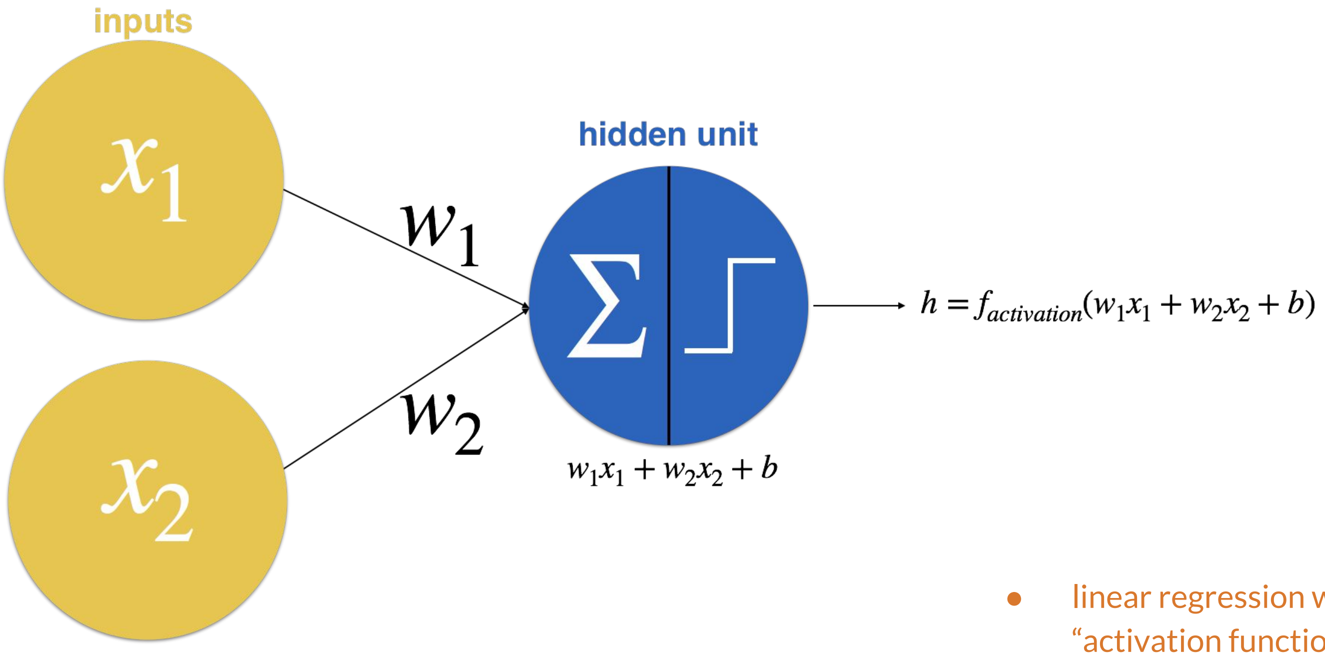


data

ANN

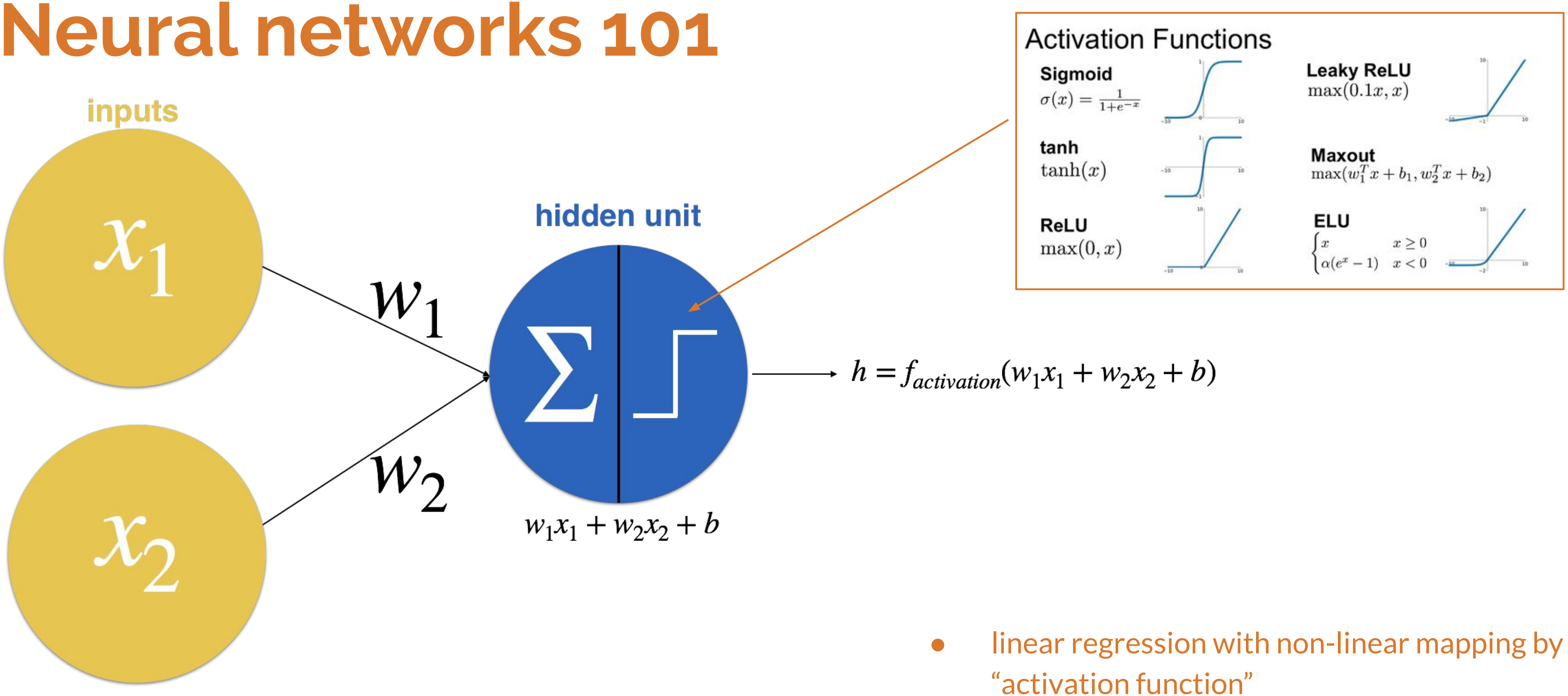
prediction

Neural networks 101



- linear regression with non-linear mapping by an "activation function"
- training of the network is merely determining the weights "w" and bias/offset "b"

Neural networks 101



Activation Functions

Sigmoid
$\sigma(x) = \frac{1}{1+e^{-x}}$

tanh
$\tanh(x)$

ReLU
$\max(0, x)$

Leaky ReLU
$\max(0.1x, x)$

Maxout
$\max(w_1^T x + b_1, w_2^T x + b_2)$

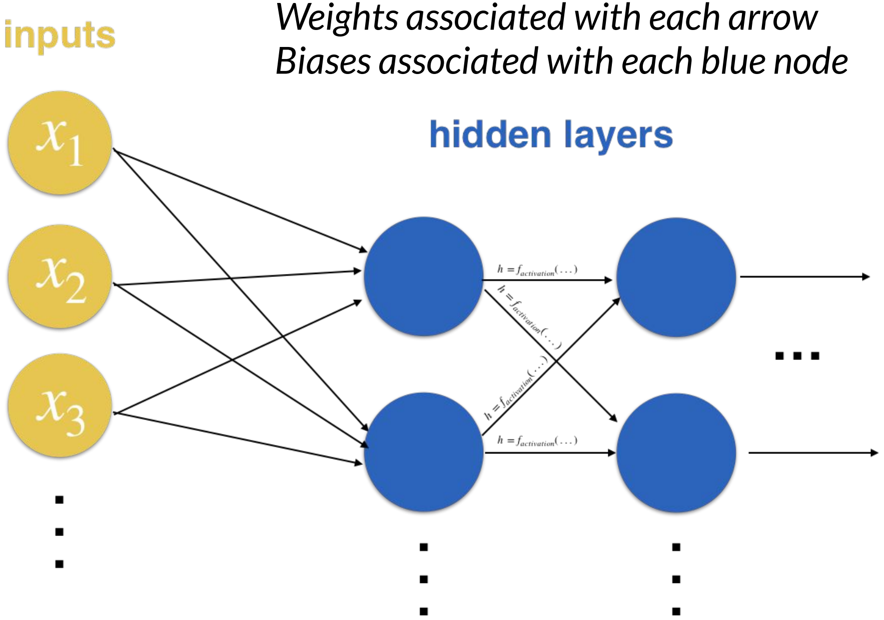
ELU
$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$

$$h = f_{activation}(w_1x_1 + w_2x_2 + b)$$

- linear regression with non-linear mapping by an "activation function"
- training of the network is merely determining the weights "w" and bias/offset "b"

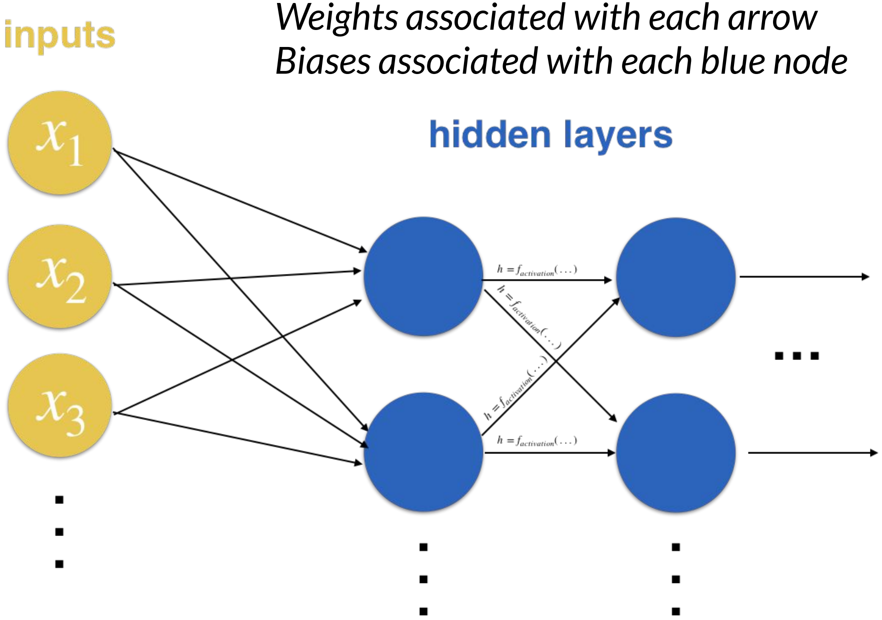
Neural networks 101

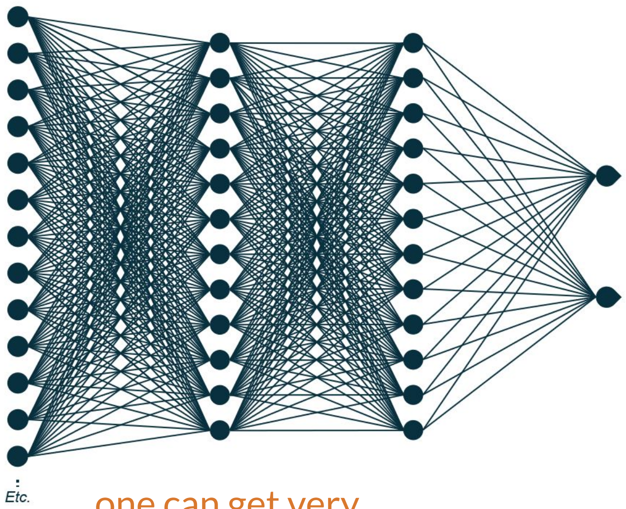
Weights associated with each arrow
Biases associated with each blue node


inputs
hidden layers
x_1
x_2
x_3
$h = f_{activation}(\ldots)$
$h = f_{activation}(\ldots)$
$h = f_{activation}(\ldots)$
$h = f_{activation}(\ldots)$

Neural networks 101

Weights associated with each arrow
Biases associated with each blue node


inputs
x_1
x_2
x_3
hidden layers
$h = f_{activation}(\ldots)$
$h = f_{activation}(\ldots)$
$h = f_{activation}(\ldots)$
$h = f_{activation}(\ldots)$

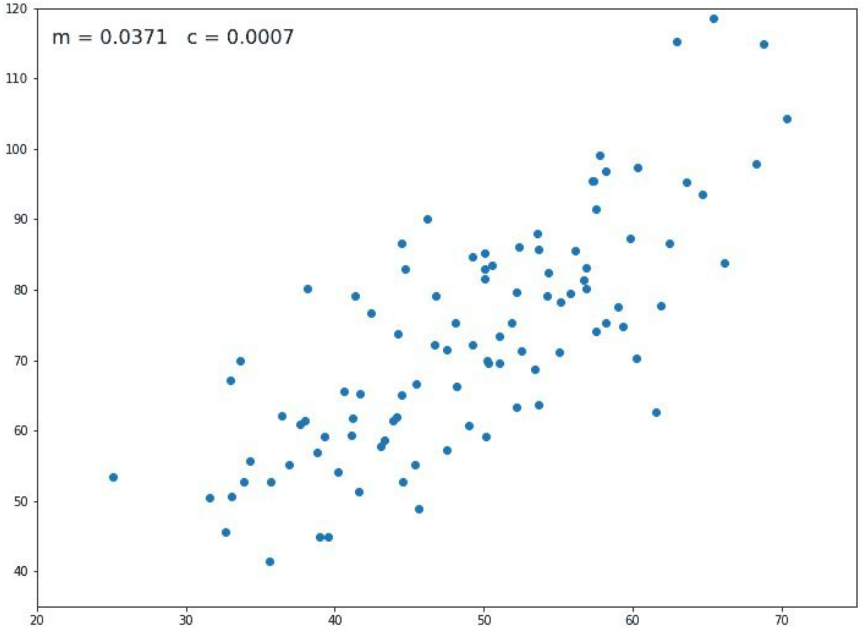


one can get very complex and “deep”

Choosing weights and biases

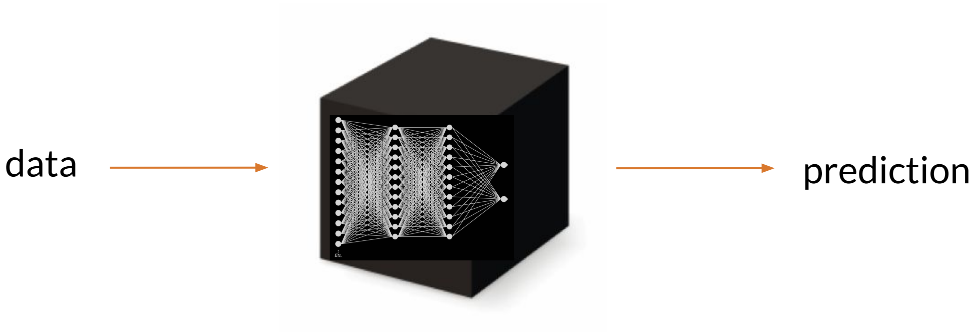
- Neural network iteratively adjusts the weights and biases using *backpropagation* and *gradient descent*
- How the error of the neural network is quantified is defined by the user
- Each incremental adjustment ideally leads to a more accurate prediction
- Iterate until the weights and biases converge

$$y_i = mx_i + b$$



Iteratively learning the best-fit weight and bias via gradient descent and backpropagation

Neural networks 101



data

prediction

Once trained, you have an array of weights and biases which can be used for prediction on new data.

Why do we care about the strategies that ANNs use?

Why care about ANN’s reasoning?

Artificial neural networks

- Have emerged as promising tool in countless NOAA-related applications.
- **Perform amazingly well at many complex tasks.**
- **ANNs are generally treated as black box**: hard to understand how they work.
- *Why is that a problem? If they work fine, why do we care how they work?*

Reason 1: Problematic strategies (Clever Hans)

Insights from a study of strategies utilized by a neural network.

Reference (also source of images on the following slides):

Lapuschkin et al. “Unmasking Clever Hans Predictors and Assessing What Machines Really Learn.” Nature Communications, vol. 10, no. 1, Mar. 2019, p. 1096, doi:10.1038/s41467-019-08987-4.

Task: Object recognition. Decide whether there is a horse in a given image.

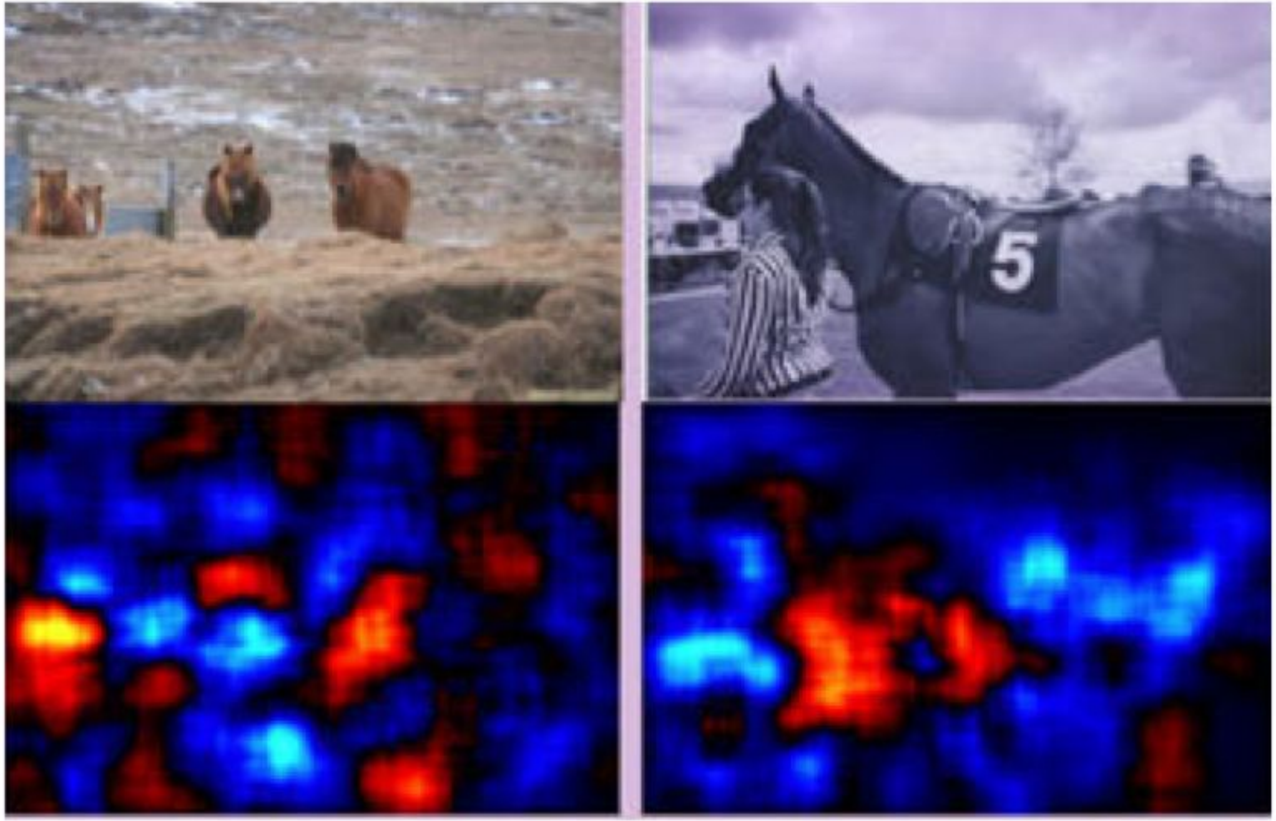
Methods used in this study:

- Neural network: to decide whether there’s a horse.
- Visualization technique (LRP): to analyze network’s strategies.

The following slides provide two things:

- An example of **problematic strategies** an ANN might use and why it might use those.
- A **way to identify such strategies**.

Input images →

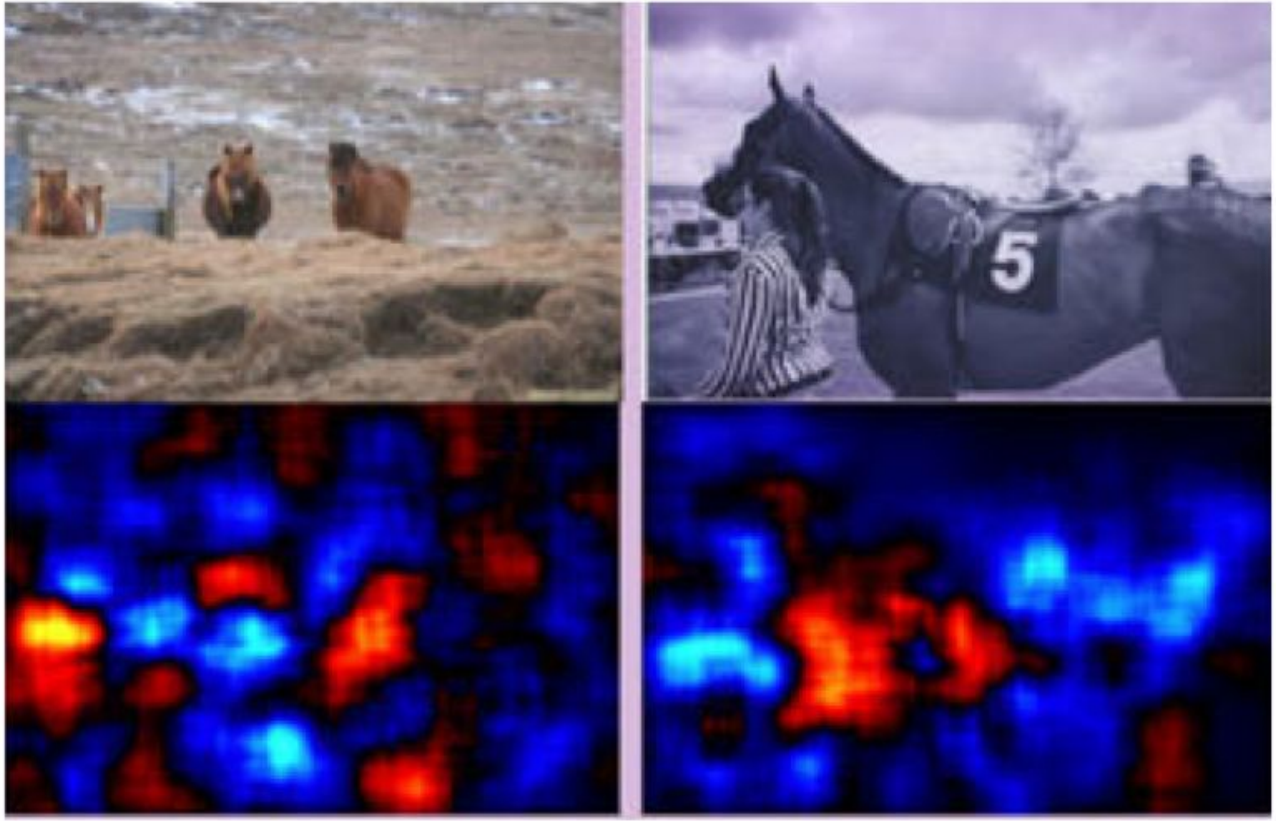


Attribution maps: Shows where the NN is looking to decide whether there is a horse.

Red areas: increase confidence
Blue areas: decrease confidence
Black areas: not useful

Strategy 1: What does the NN detect?

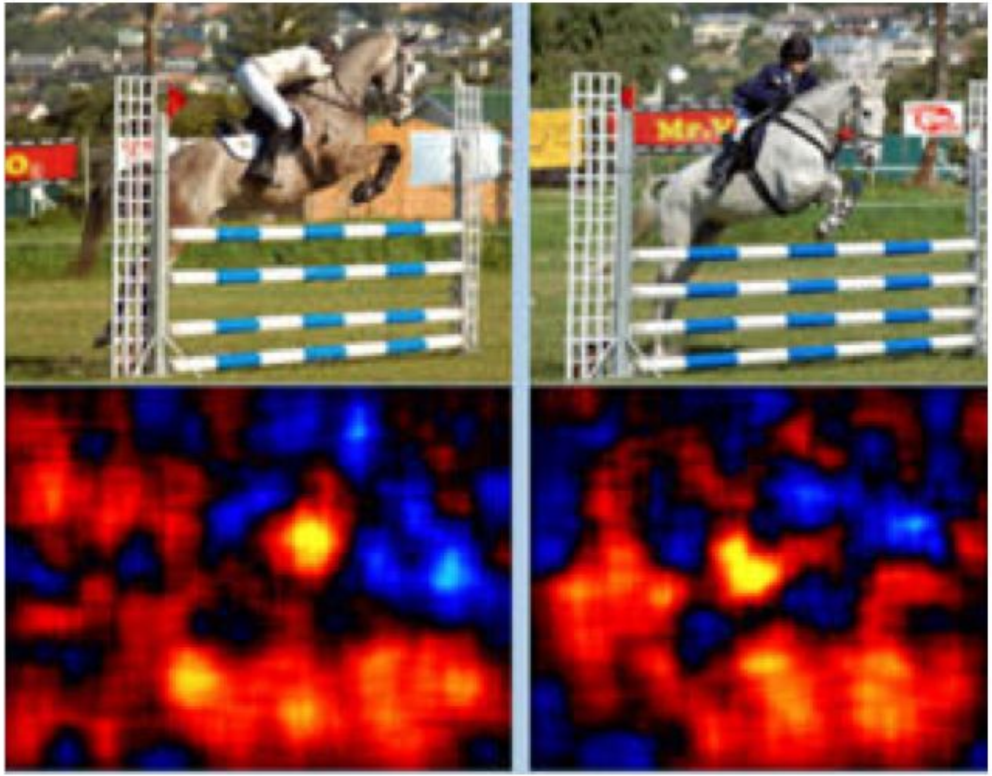
Input images →



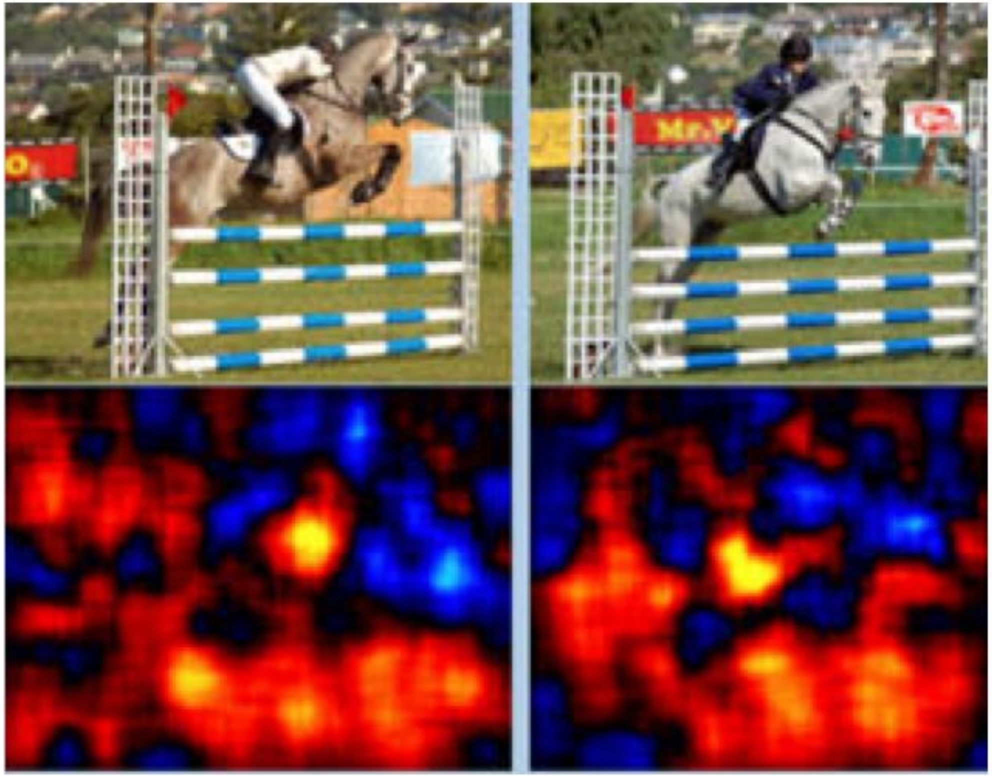
Attribution maps: Shows where the NN is looking to decide whether there is a horse.

Red areas: increase confidence
Blue areas: decrease confidence
Black areas: not useful

Strategy 1: Detects the pixels with horses. Good!

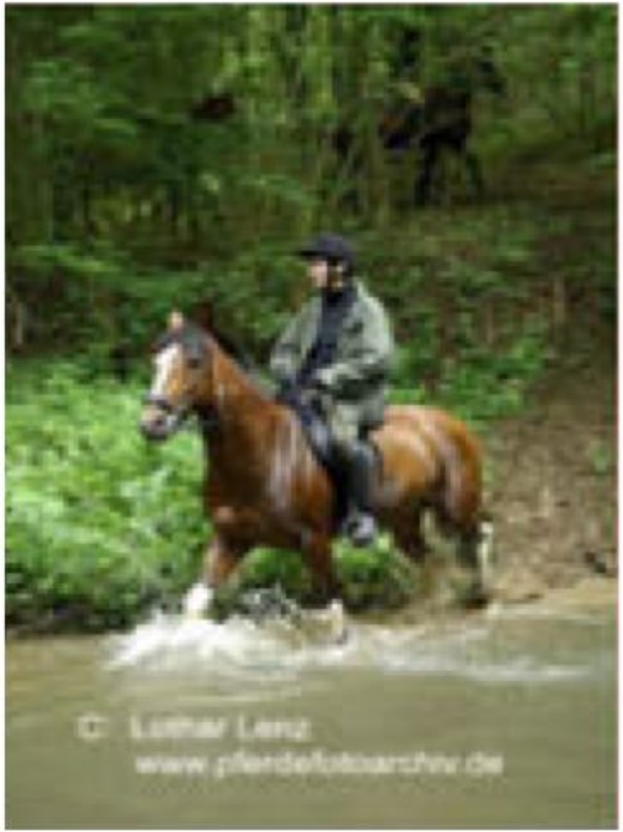


Strategy 2: What does the NN detect?

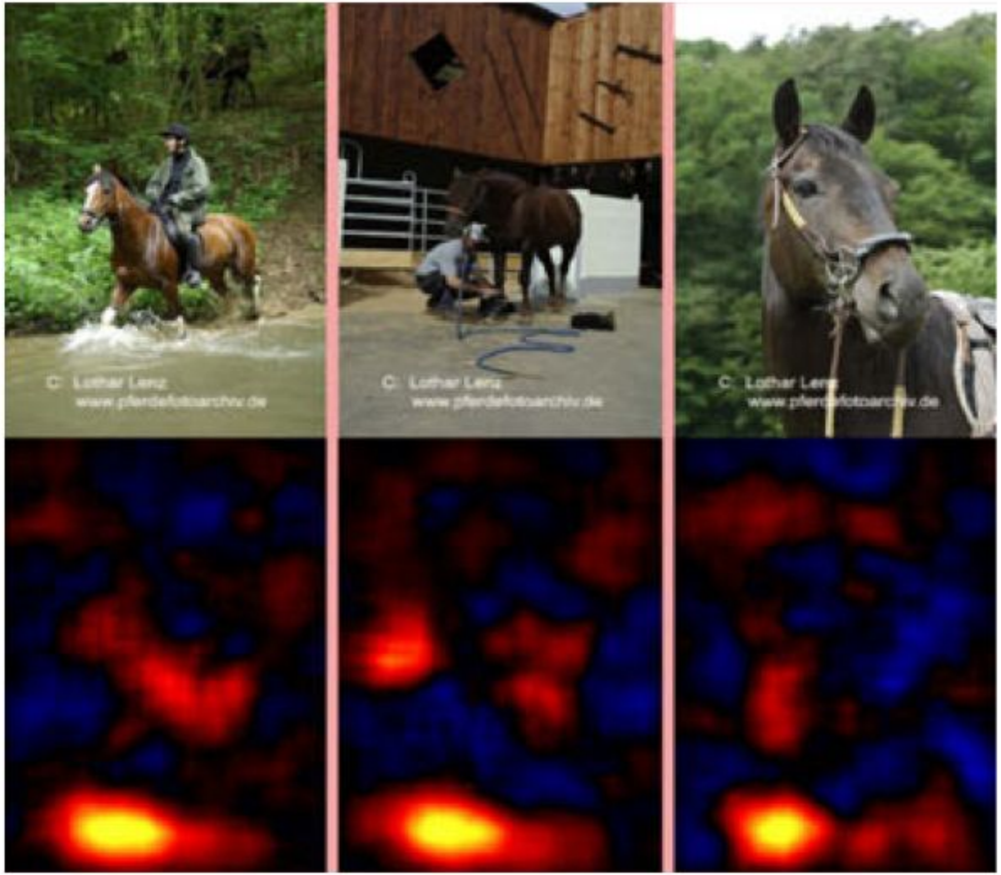


Strategy 2: **Detects correlated objects: poles.**

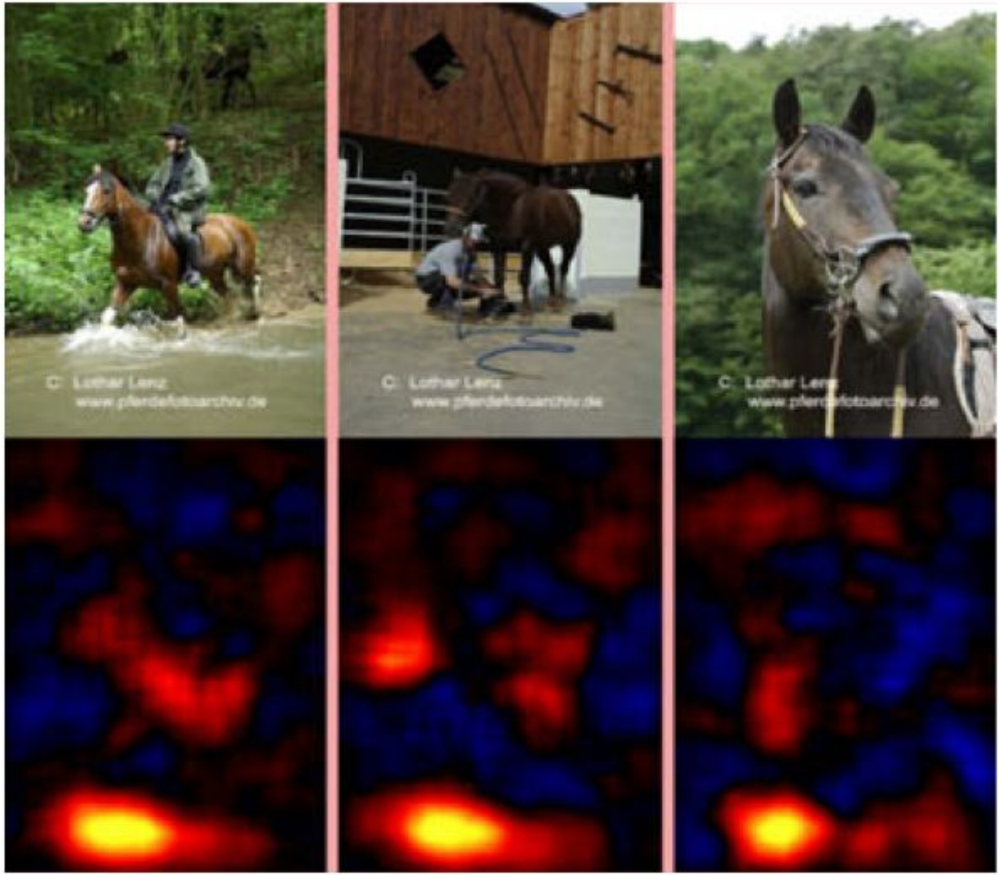
Acceptable? Depends on where this will be used.



Strategy 3: What does it detect?

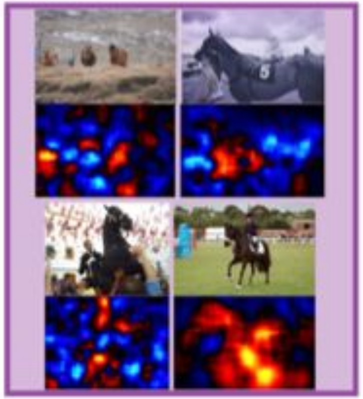


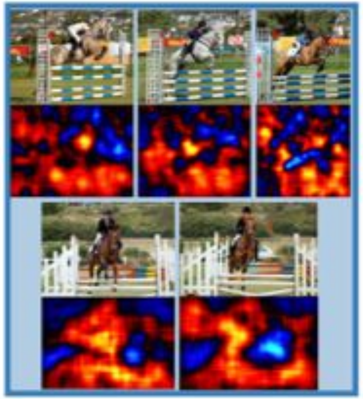
Strategy 3: What does it detect?

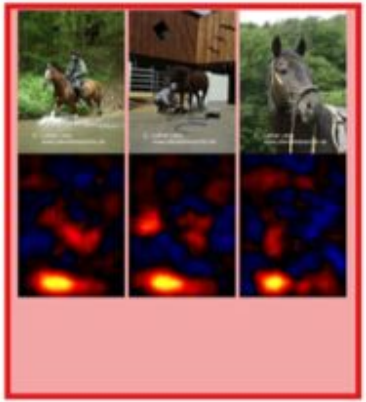


Strategy 3: **Detects html tags!** Not acceptable!

ANN used three different strategies







ANN detects: **horses (great!)**

poles
(might be acceptable)

html tags
(not accetable!)

Correlated objects: Might or might not be acceptable

Questions:

- Which strategies do you actually want?
- Which ones of these strategies will work in the "real world"?

What to learn from this example

- Algorithm correctly learned ***correlations present in the data****.*
- But some of the correlations were not representative of correlations in real world (e.g., html tags).
  → **Algorithm seems to perform well, but learned reasoning that does not generalize to the world.**
- **Conclusion: Using ANN as black box is a problem in this case.**

Other problems when using ANNs as a black box:

- Earth scientists working with these tools might be alienated by lack of understanding - and rightly so → **no trust**.
- Primary way of improving ANNs: trial-and-error. Wouldn't it be nice to have guiding tools on how to improve them?
- Does not encourage physics-guided machine learning.

How visualization methods can help

Using visualization tools can provide:

- Provide information on ANN’s reasoning (see above examples), e.g., in form of attribution maps, as shown above.

In turn that provides:

1. Important information for design of ANNs.
2. Increases trust, encourages physics-guided machine learning.
3. **Provides new role for ML:** visualization output can even be used **to discover *new science!*** *(See examples in last part of this tutorial)*.

What to expect from visualization

Put backpack into X ray scanner

Inside view

Scanning backpack through Xray scanner.

Not a perfect view, but a lot better than just looking at the outside.

What to expect from visualization

Put backpack into X ray scanner →

Inside view →

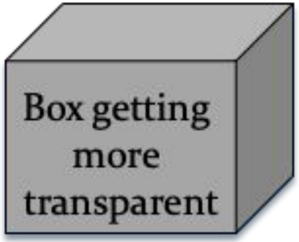
Comparison:

Black Box

Put box into tool →

Tools for visualization + interpretation of ML methods

Inside view →



Box getting more transparent

Sample tools for neural networks:

Heat maps, feature visualization, visualization for convolution networks.

Background for ANN visualization tools

Related Work - Recent Article

Recent article - written for climate/weather community**:**

McGovern A, Lagerquist R, Gagne DJ, Jergensen GE, Elmore KL, Homeyer CR, Smith T. **Making the black box more transparent: Understanding the physical implications of machine learning**. *Bulletin of the American Meteorological Society*. **Aug 22, 2019**. https://journals.ametsoc.org/doi/abs/10.1175/BAMS-D-18-0195.1

Provides:

1. **Overview of general ML interpretation/visualization methods.**
2. **Specifically for ANNs:**
    - Saliency maps (discussed below)
    - Backwards Optimization (discussed below)
    - Gradient-weighted Class-activation Maps
    - Novelty Detection
3. **Demonstration for applications:**

    Storm-mode, precipitation type, tornado prediction, and hail prediction.

Related Work - Explainable AI Book

Recent book on Explainable AI - *not* specific to climate/weather:

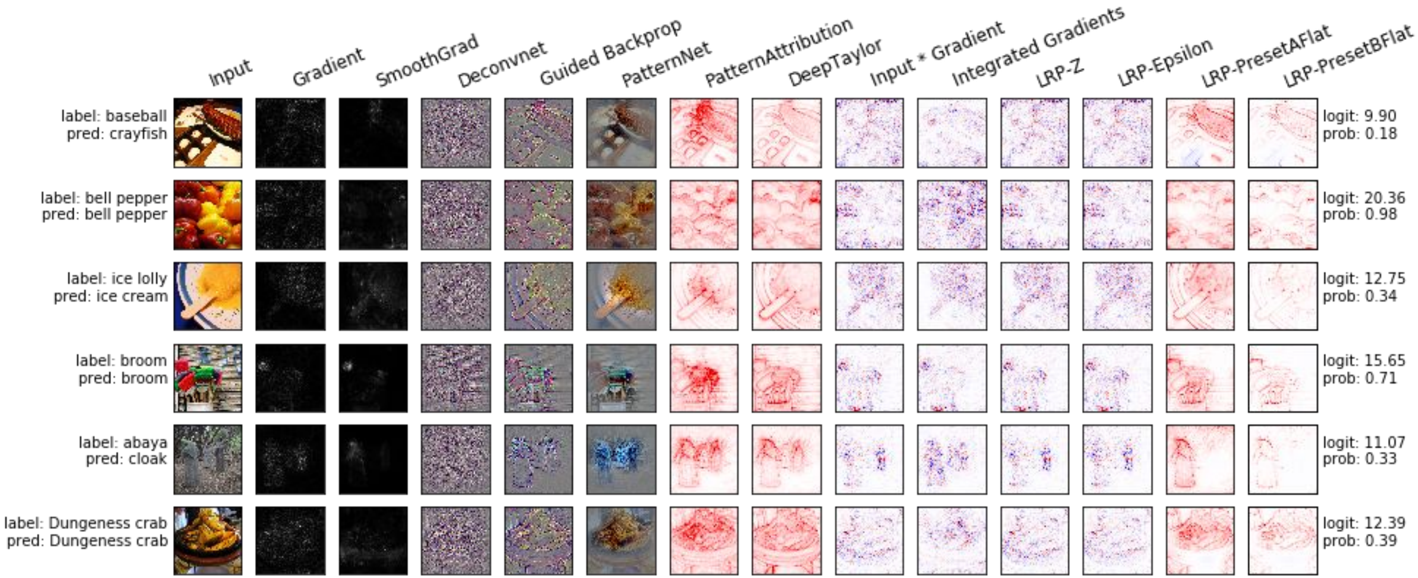
Samek, W., Montavon, G., Vedaldi, A., Hansen, L.K., Muller, K.-R.,
Explainable AI: Interpreting, Explaining and Visualizing Deep Learning.
Springer Nature, **Aug 30, 2019.**
https://www.springer.com/gp/book/9783030289539.

Provides:

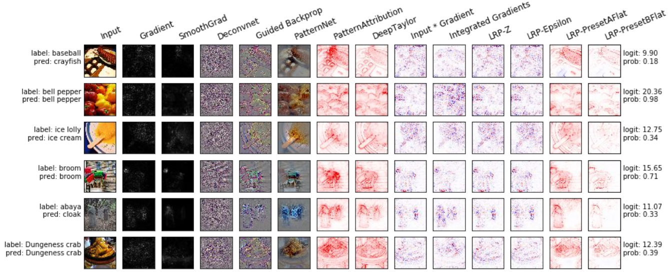
- General overview of interpretation and visualization methods.
- Primarily for ANNs.
- 439 pages.

Related Work - Implementations

- Many methods from the book are already implemented in toolboxes.
- Example: “Innvestigate” toolbox for Keras, available at https://github.com/albermax/innvestigate



Our Preferred Method - as of now



Our favorite method:

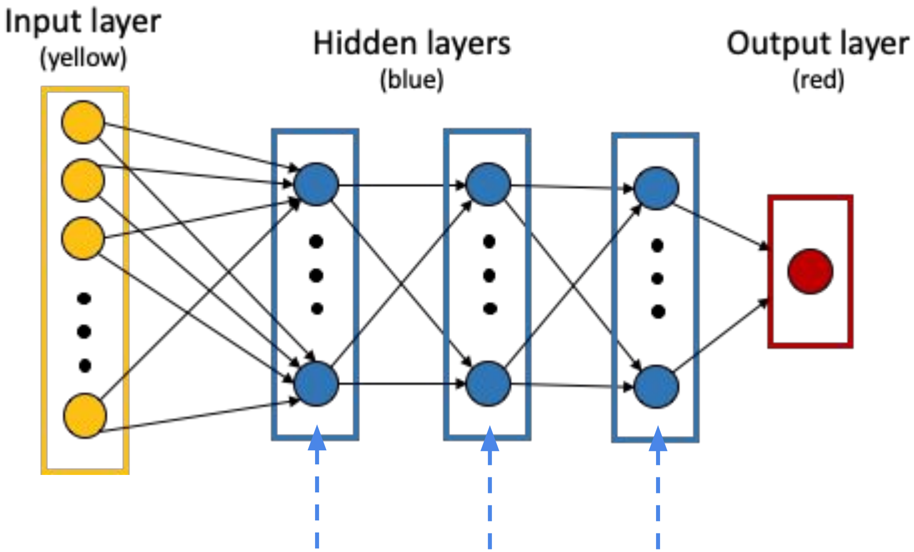
- Layer-wise relevance propagation (LRP).
- Has not yet been used in climate/weather.
- We believe LRP is particularly powerful.
- Focus of this presentation.

- But: New methods are being developed as we speak.
- **The purpose of this tutorial**
  - ***Is not* to promote LRP as “the best method”.**
  - **Is to show what visualization methods in general can do for the community!**

Two types of visualization tools

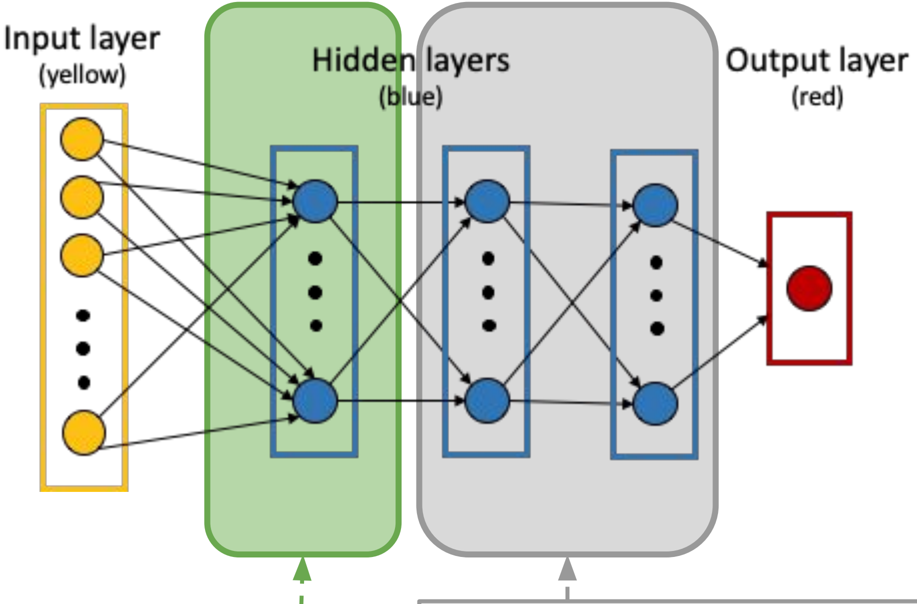
Type A: Feature Visualization

Philosophy: Seek to understand all internal components of ANN.



Seek to understand the meaning of all intermediate (blue) nodes.

Type A: Feature Visualization



First hidden layer:

- Special case.
- Very easy to interpret, because directly related to input space.
- Often useful to visualize ANN weights for first layer.
- Easy to do. Easy to interpret.

Later layers:

- Much more abstract.
- Particularly hard to interpret for ANNs trained on objects with fuzzy boundaries.

Type A: Feature Visualization

Method for visualizing first layer:

- For each node in first layer:
    - Show input image corresponding to first layer weights of that neuron.
    - Tells you which input image each neuron represents. Done!

Method for later (deeper) layers:

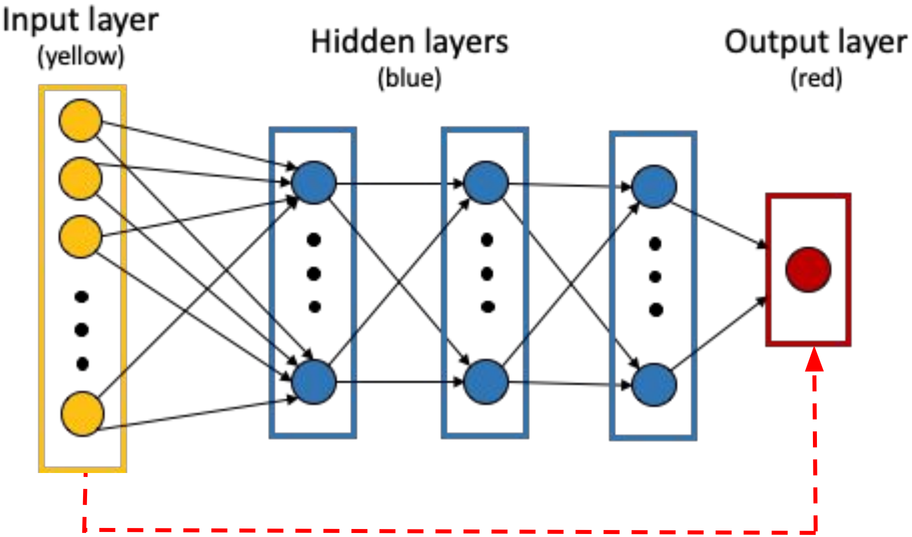
- **Common Method**:
    - Generate some kind of “image representation” for any specific neuron as follows.
    - Optimize input to maximally trigger the specific neuron (“Backwards Optimization”)
    - Easy to do - uses built-in back propagation mechanics of ANN.
    - But **high-frequency artifacts** often occur - especially when using strided convolution & pooling (creates high frequency component in gradient).
    - Hard to interpret.

Comment:

- *For our applications*, we have sometimes found it worthwhile to visualize weights of first hidden layers, but not of deeper layers.

Type B: Attribution / Explaining Decisions

Philosophy: Understand the ANN's overall decision making for specific input.



Seek to understand the meaning of the entire algorithm - for a specific input.
Do NOT worry about meaning of intermediate (blue) nodes.

Type B: Attribution

Goal: Given a specific input image, understand how ANN comes up with the output, e.g., a predicted class. (Primarily developed for classification - more on that later.)

Method:

- Generates “Attribution map” - also known as “Heat map”
- “Map” is a computer science term here - has nothing to do with a geographic map.
- Map = overlay for input
- Input does not have to be an image, but images used her for simplicity.
- Given input, highlight which parts of input are important to lead to corresponding output.

→ Seeing which part of input is most important for a specific class can provide intuition of network’s strategies.

We find Type B methods much more useful for our applications. Focus from here on.

Type B Methods briefly discussed here

1) **Perturbation (occlusion) methods**
   Reason for including: intuitive, baseline method.

2) **Saliency**
   Reason for including: very common.

3) **Layer-wise relevance propagation (LRP)**
   Reason for including: Currently, our preferred method for our applications.

- Other methods exist. New methods are being developed as we speak!

All three methods produce “heat maps” with color code:

- Red: area makes output class more likely.
- Blue: area makes output class less likely.
- Black (or white): area not important for output.

Set-up considered

General set-up discussed here:

1. Train ANN model using training data.
2. Freeze ANN parameters.
3. Seek to understand reasoning of frozen ANN model.

For ease of explanation:

- Focus on “image” type input here.
- Output: predicted *class* (discrete).

Example yielding a class (“classification task”):

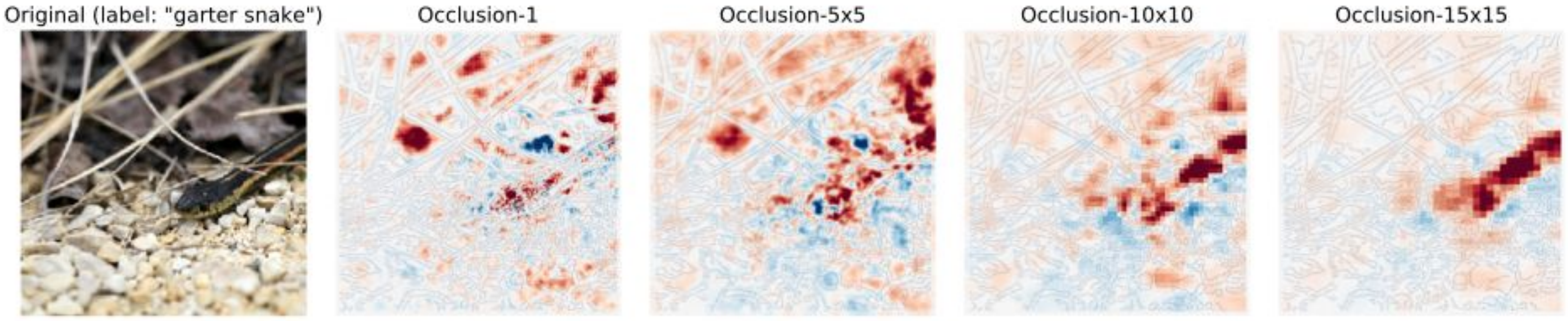
- Is there a hurricane in this satellite image (0/1)?
- In which phase (1-8, or none) is MJO in this image?

Method 1: Perturbation / Occlusion

Idea:

- Cover one part of image at a time.
- See how this changes the output.

Example: Detect whether there is a snake in the input image.
Source: Ancona, Marco, et al. "Towards Better Understanding of Gradient-Based Attribution Methods for Deep Neural Networks." ICLR 2018, arxiv.org, 2018, http://arxiv.org/abs/1711.06104.



Observation: Occlusion method works here **only if the occlusion patch size is large enough to cover most of the object being detected**.

Nevertheless: intuitive baseline method. Just make sure to **choose occluded patch big enough**.

Method 2: Saliency (Gradient)

Idea: Calculate H = gradient of neuron activation over input pixels

$$\mathbf{H} = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = \frac{\partial \text{ output neuron value}}{\partial \textbf{ input values}}$$

ANNs are gradient-based methods

→ Easy to calculate H with built-in mechanics of ANN.

→ Yields heat map H.

→ H tells you: *How should we change input to maximally increase/decrease output?*

Saliency maps:

- Widely available method.
- Applicable for both classification and regression.

Method 2: Saliency - Interpretation

Saliency answers this question:

What should we change in input to maximally increase/decrease output value?

Analogy: Which change would maximally increase/decrease Warren Buffet’s wealth?

Question we are usually more interested in for our applications:

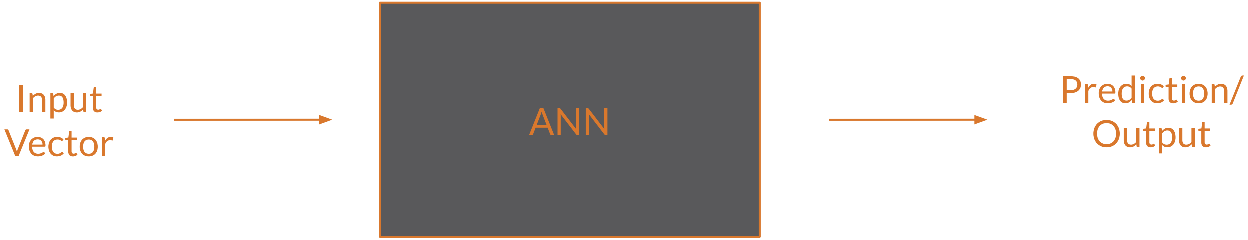
What in the input led to the current output value?

Analogy: What made Warren Buffet so rich in the first place? What contributed most to his wealth?

→ To answer *the latter* question:
→ **Method 3: Layer-wise relevance propagation (LRP).**
→ See next section.

Layer-wise Relevance Propagation

Layer-wise Relevance Propagation [LRP]

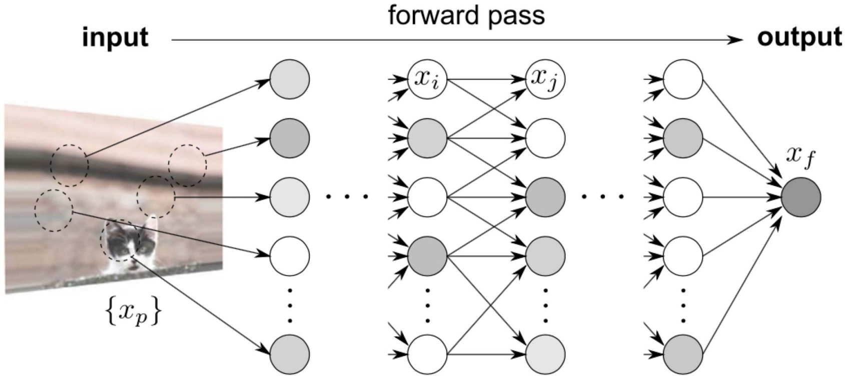


Which part(s) of the specific input were most **relevant** for the ANN's prediction?

- Used after the model has been trained
- Output relevance is computed separately for each input

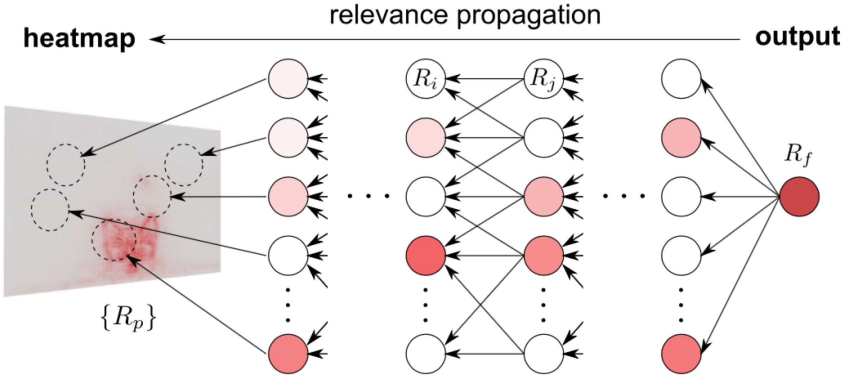
Layer-wise Relevance Propagation [LRP]

Prediction



Probability of CAT

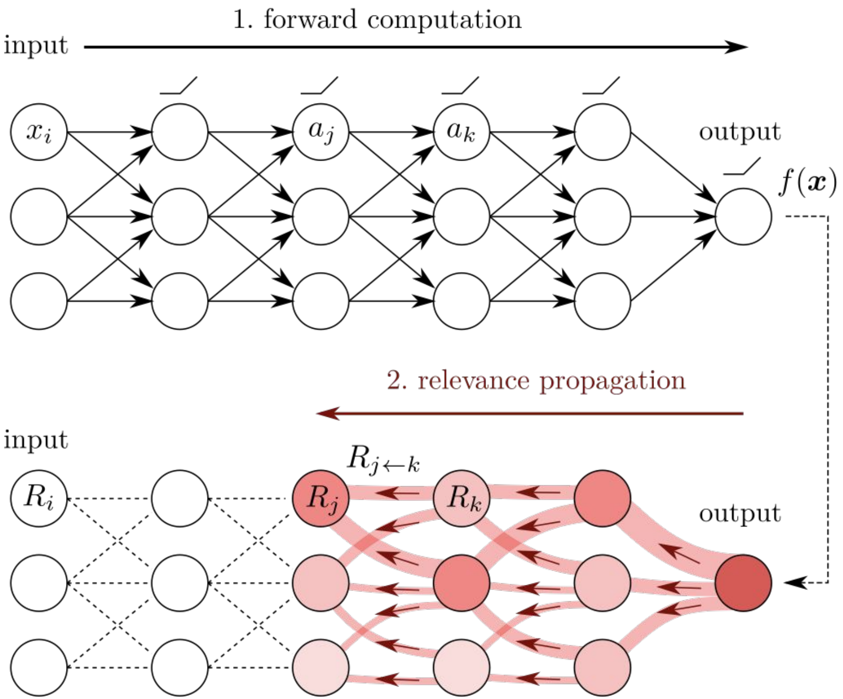
LRP

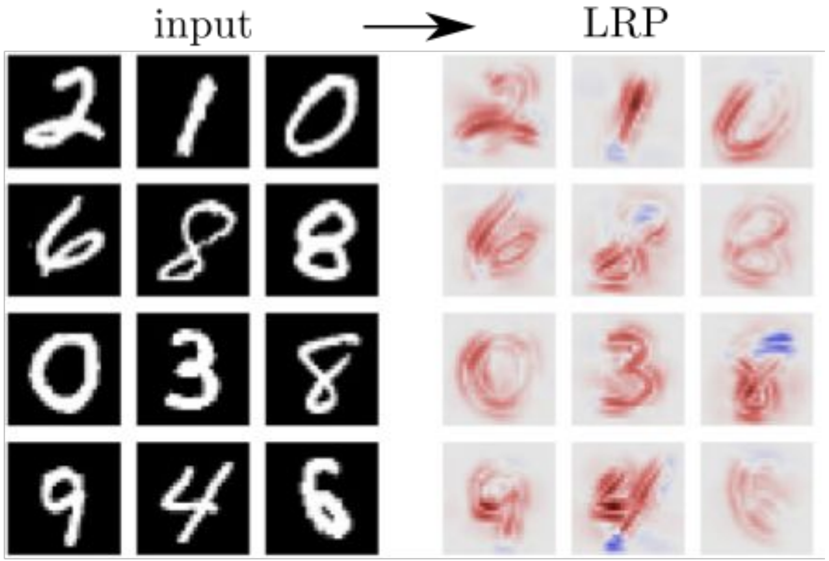


Probability of CAT

Montavon et al. (2017)

How it works



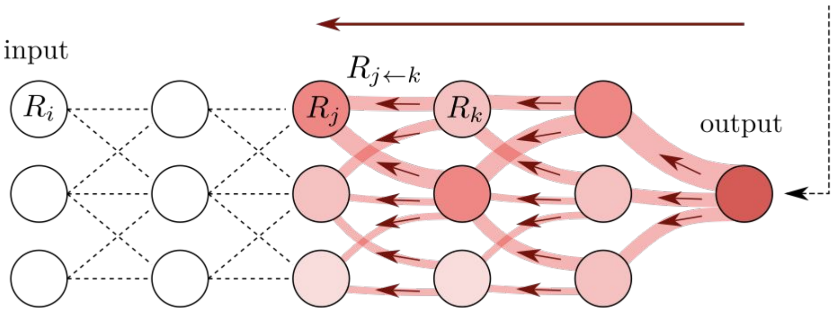


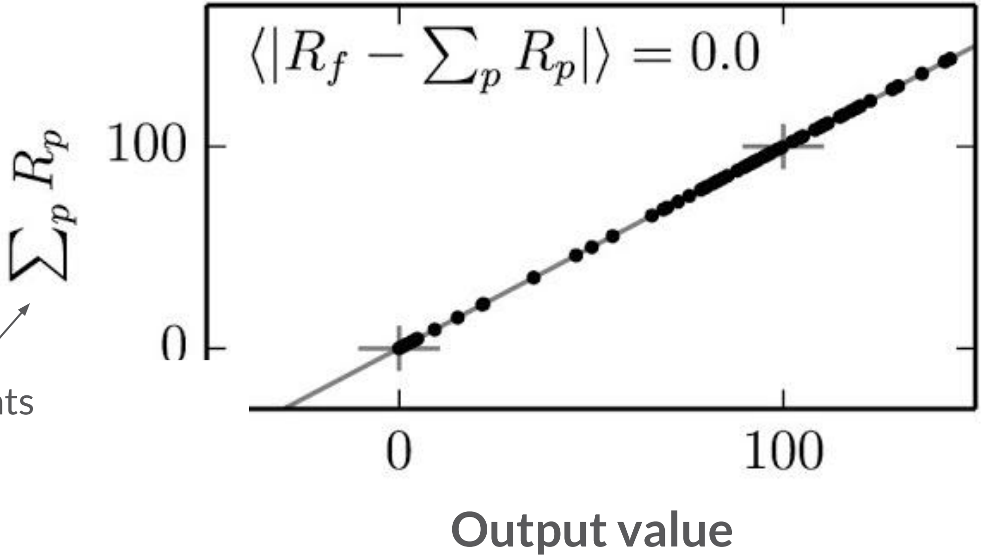
Conserves the output during propagation

Montavon et al. (2018)
method described in Toms et al. (in prep)

LRP preserves information from the output to input

- Dots along the line show conservation of information from the output to the input
- Rules have been constructed such that the information is conserved
- Ensures that all regions of relevance are captured by the heatmap

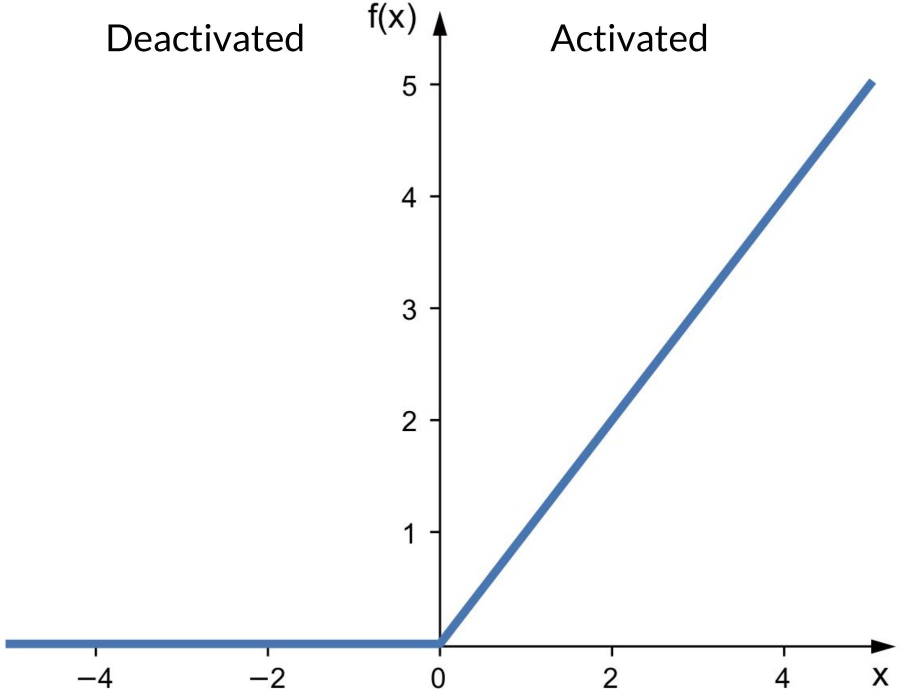




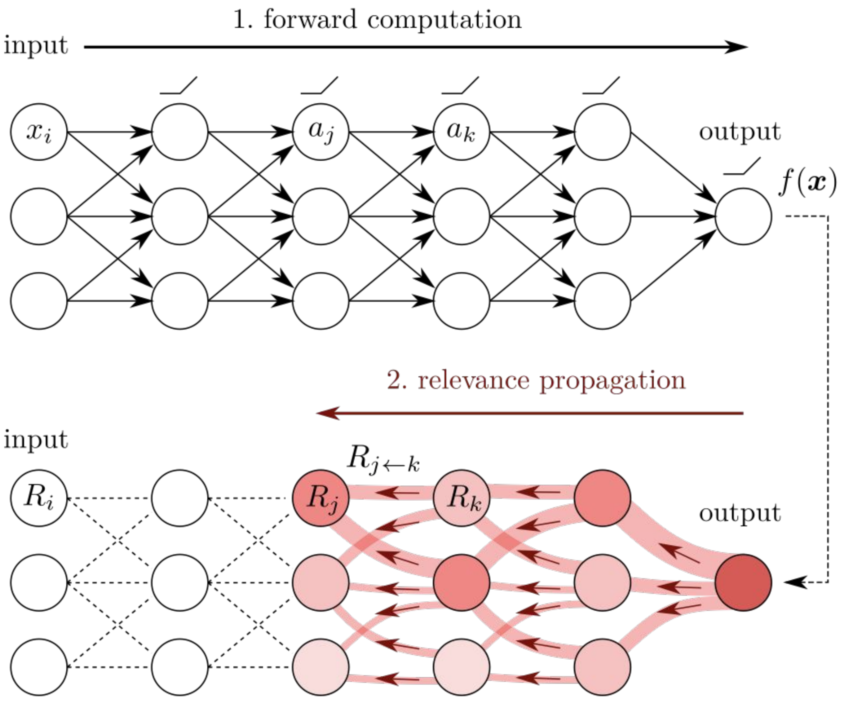
sum over weights in one layer

LRP developed for ReLU to show activation vs deactivation

- Positive values show activation, while negative values are deactivated and are ignored
- Only activated outputs from neurons are propagated backwards
- Similar to how only activated outputs from neurons are propagated forwards during training



Alpha-Beta Rule

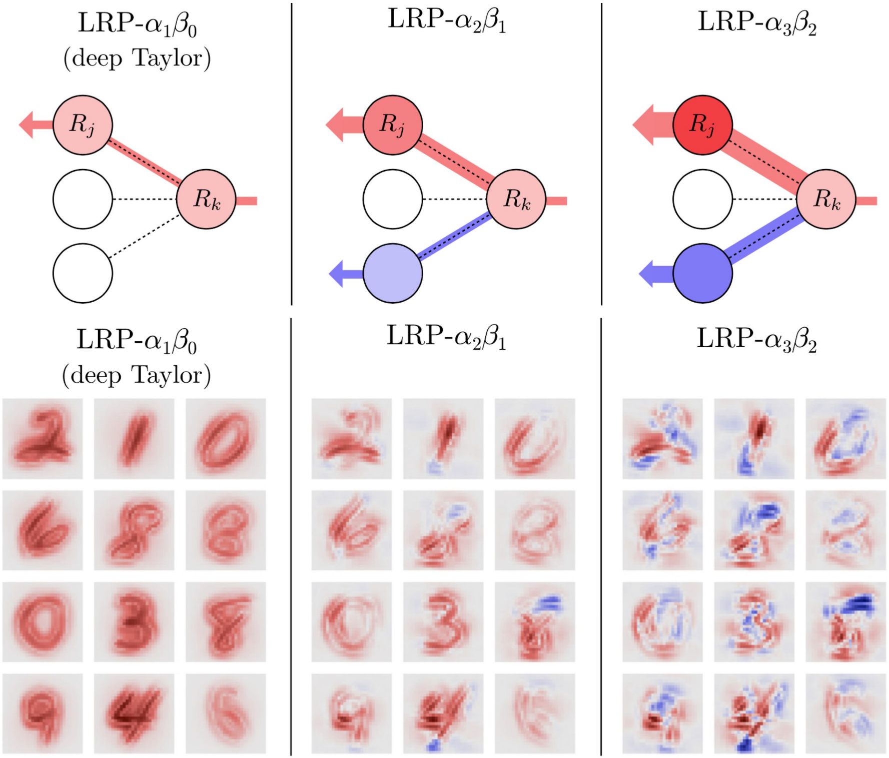


tunable parameters: α, β
fixed parameters: *a*, *w*, *R*

$$R_j = \sum_k \left(\alpha \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} - \beta \frac{a_j w_{jk}^-}{\sum_j a_j w_{jk}^-} \right) R_k$$

Montavon et al. (2018)
method described in Toms et al. (in prep)

Alpha-Beta Rule



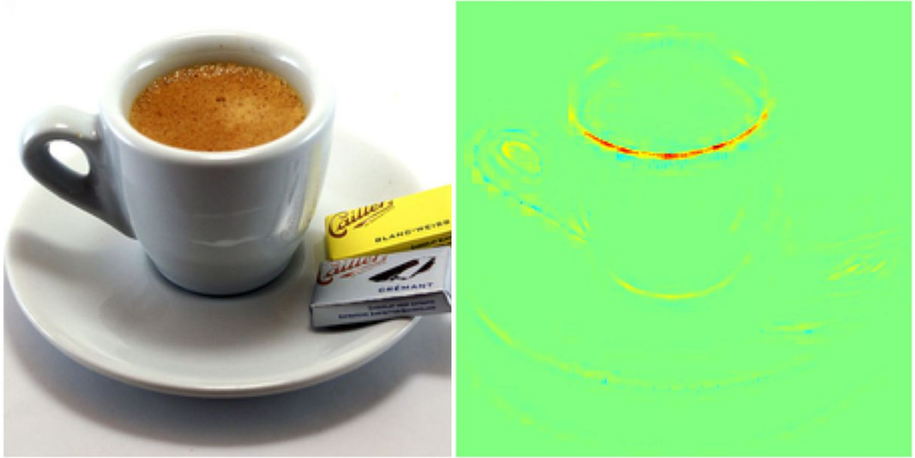
tunable parameters: α, β
fixed parameters: *a*, *w*, *R*

$$R_j = \sum_k \left(\alpha \frac{a_j w_{jk}^+}{\sum_j a_j w_{jk}^+} - \beta \frac{a_j w_{jk}^-}{\sum_j a_j w_{jk}^-} \right) R_k$$

Montavon et al. (2018)
method described in Toms et al. (in prep)

A few more LRP examples

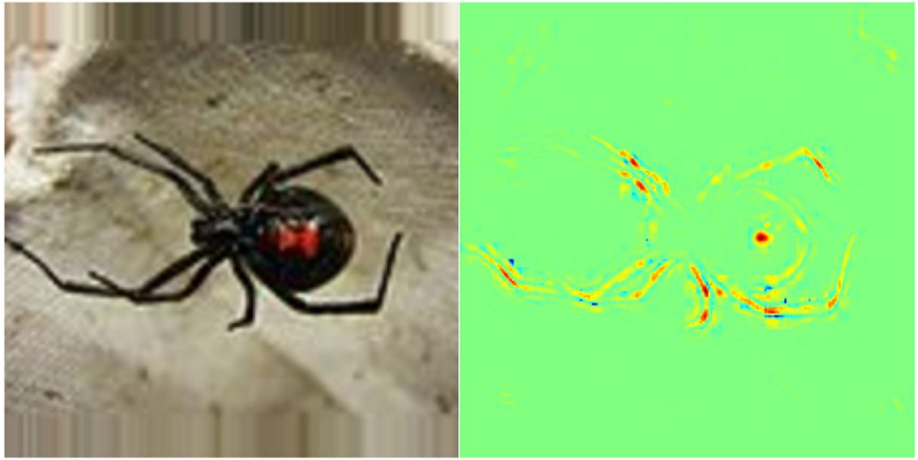
alpha=2
beta = 1



Bach et al. (2015)

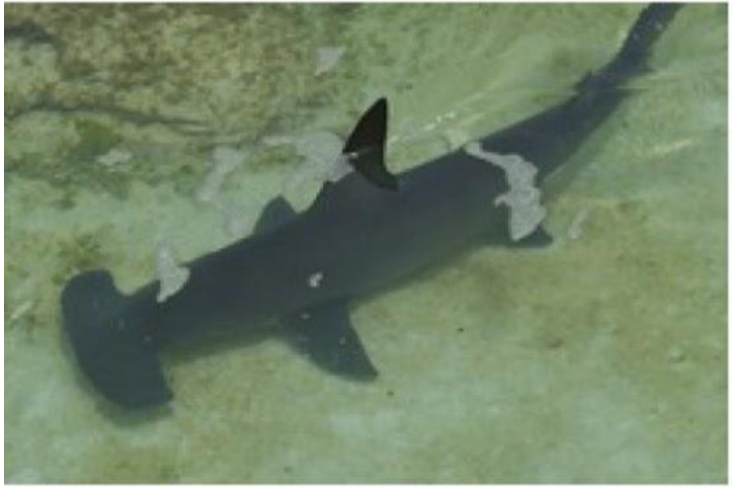
A few more LRP examples

alpha=2
beta = 1

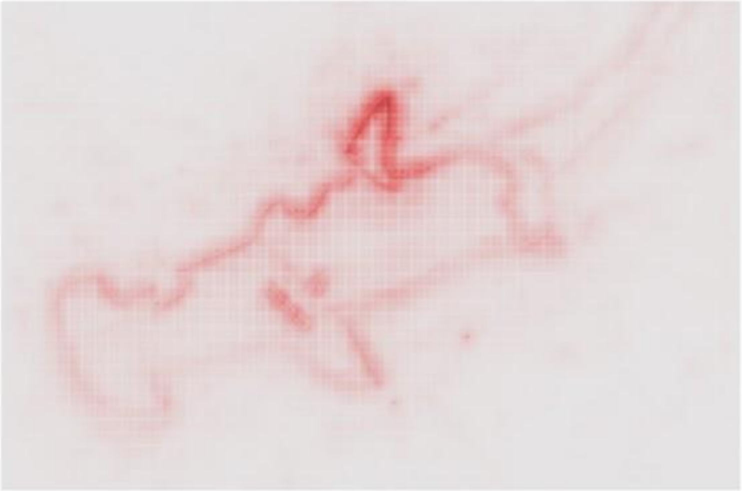


Bach et al. (2015)

A few more LRP examples



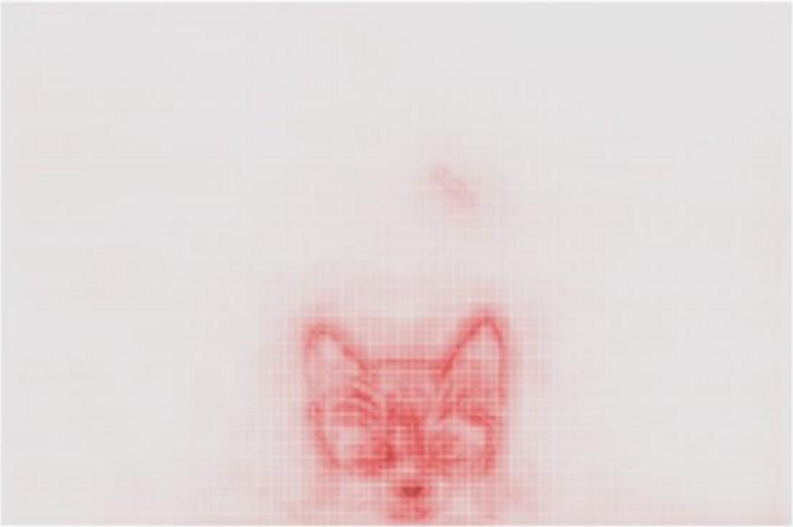
alpha = 1
beta = 0



Montavon et al. (2017)

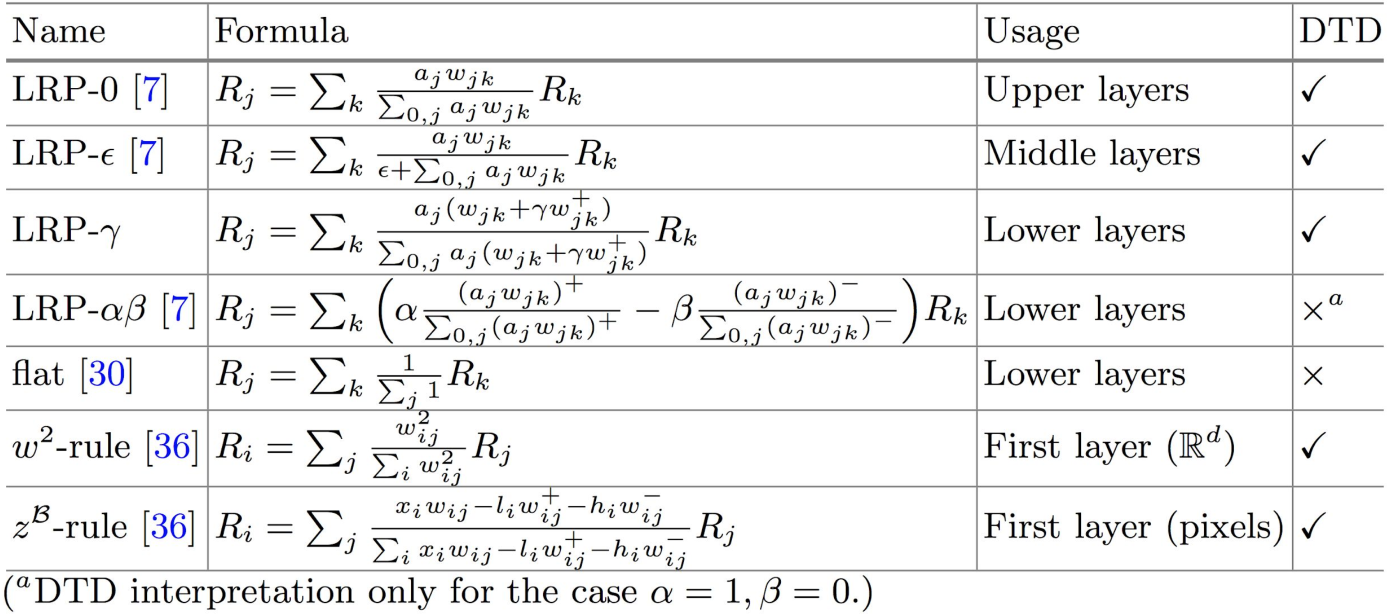
A few more LRP examples

alpha = 1
beta = 0



Montavon et al. (2017)

Many other “rules” to explore



Name	Formula	Usage	DTD
LRP-0 [7]	$R_j = \sum_k \frac{a_j w_{jk}}{\sum_{0,j} a_j w_{jk}} R_k$	Upper layers	✓
LRP-ϵ [7]	$R_j = \sum_k \frac{a_j w_{jk}}{\epsilon + \sum_{0,j} a_j w_{jk}} R_k$	Middle layers	✓
LRP-γ	$R_j = \sum_k \frac{a_j (w_{jk} + \gamma w_{jk}^+)}{\sum_{0,j} a_j (w_{jk} + \gamma w_{jk}^+)} R_k$	Lower layers	✓
LRP-$\alpha\beta$ [7]	$R_j = \sum_k \left(\alpha \frac{(a_j w_{jk})^+}{\sum_{0,j} (a_j w_{jk})^+} - \beta \frac{(a_j w_{jk})^-}{\sum_{0,j} (a_j w_{jk})^-} \right) R_k$	Lower layers	$\times^a$
flat [30]	$R_j = \sum_k \frac{1}{\sum_j 1} R_k$	Lower layers	×
w^2-rule [36]	$R_i = \sum_j \frac{w_{ij}^2}{\sum_i w_{ij}^2} R_j$	First layer ($\mathbb{R}^d$)	✓
$z^{\mathcal{B}}$-rule [36]	$R_i = \sum_j \frac{x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-}{\sum_i x_i w_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-} R_j$	First layer (pixels)	✓

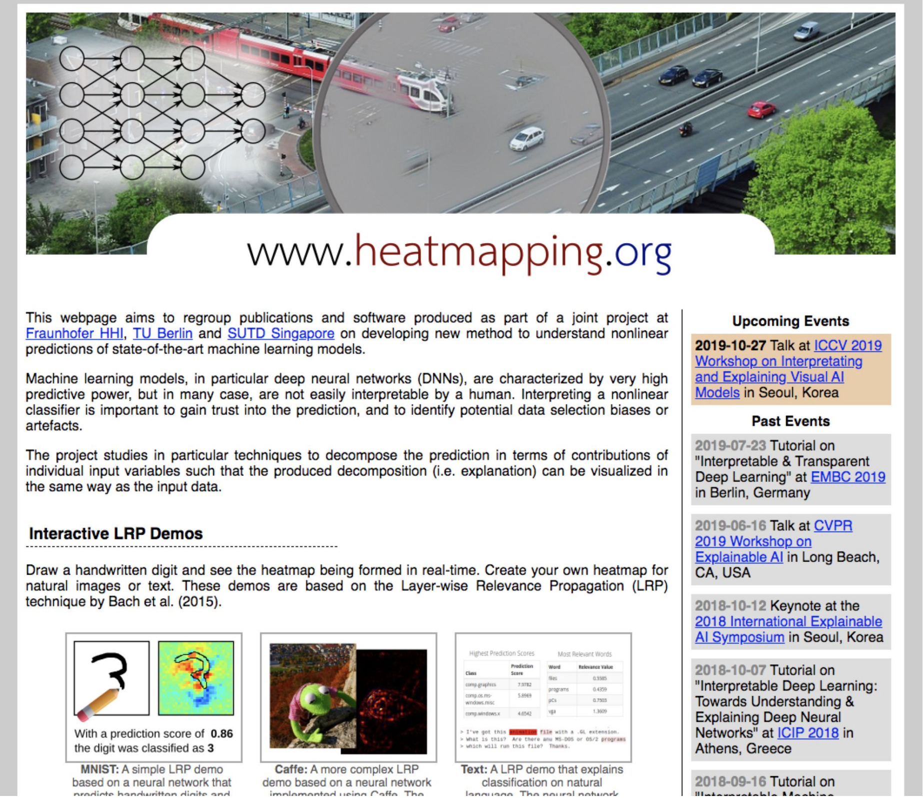
(^{a}DTD interpretation only for the case $\alpha = 1, \beta = 0$.)

Colorado State University

Samek et al. (2019)

heatmapping.org

- investigate package
  - lots of options
  - not perfect, but a useful tool for our group



This webpage aims to regroup publications and software produced as part of a joint project at Fraunhofer HHI, TU Berlin and SUTD Singapore on developing new method to understand nonlinear predictions of state-of-the-art machine learning models.

Machine learning models, in particular deep neural networks (DNNs), are characterized by very high predictive power, but in many case, are not easily interpretable by a human. Interpreting a nonlinear classifier is important to gain trust into the prediction, and to identify potential data selection biases or artefacts.

The project studies in particular techniques to decompose the prediction in terms of contributions of individual input variables such that the produced decomposition (i.e. explanation) can be visualized in the same way as the input data.

Interactive LRP Demos

Draw a handwritten digit and see the heatmap being formed in real-time. Create your own heatmap for natural images or text. These demos are based on the Layer-wise Relevance Propagation (LRP) technique by Bach et al. (2015).

With a prediction score of **0.86** the digit was classified as **3**

MNIST: A simple LRP demo based on a neural network that predicts handwritten digits and

Caffe: A more complex LRP demo based on a neural network implemented using Caffe. The

Highest Prediction Scores

Class	Prediction Score
comp.graphics	7.9782
comp.os.ms-windows.misc	5.8969
comp.windows.x	4.6542

Most Relevant Words

Word	Relevance Value
files	0.5585
programs	0.4359
pCs	0.7503
vga	1.3609

```
> I've got this animation file with a .GL extension.
> What is this?  Are there anu MS-DOS or OS/2 programs
> which will run this file?  Thanks.
```

Text: A LRP demo that explains classification on natural language. The neural network

Upcoming Events

2019-10-27 Talk at ICCV 2019 Workshop on Interpreting and Explaining Visual AI Models in Seoul, Korea

Past Events

2019-07-23 Tutorial on "Interpretable & Transparent Deep Learning" at EMBC 2019 in Berlin, Germany

2019-06-16 Talk at CVPR 2019 Workshop on Explainable AI in Long Beach, CA, USA

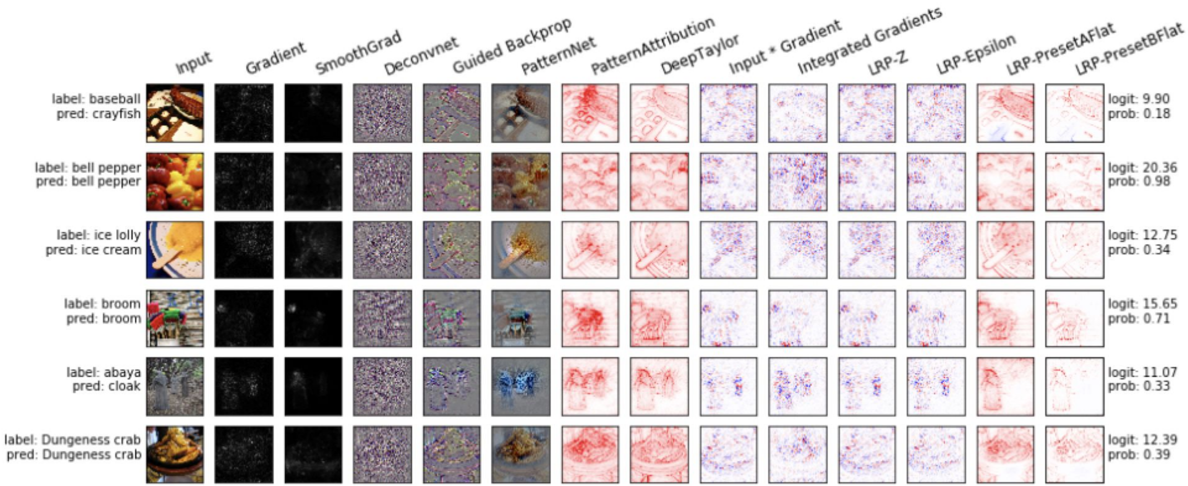
2018-10-12 Keynote at the 2018 International Explainable AI Symposium in Seoul, Korea

2018-10-07 Tutorial on "Interpretable Deep Learning: Towards Understanding & Explaining Deep Neural Networks" at ICIP 2018 in Athens, Greece

2018-09-16 Tutorial on "Interpretable Machine

heatmapping.org

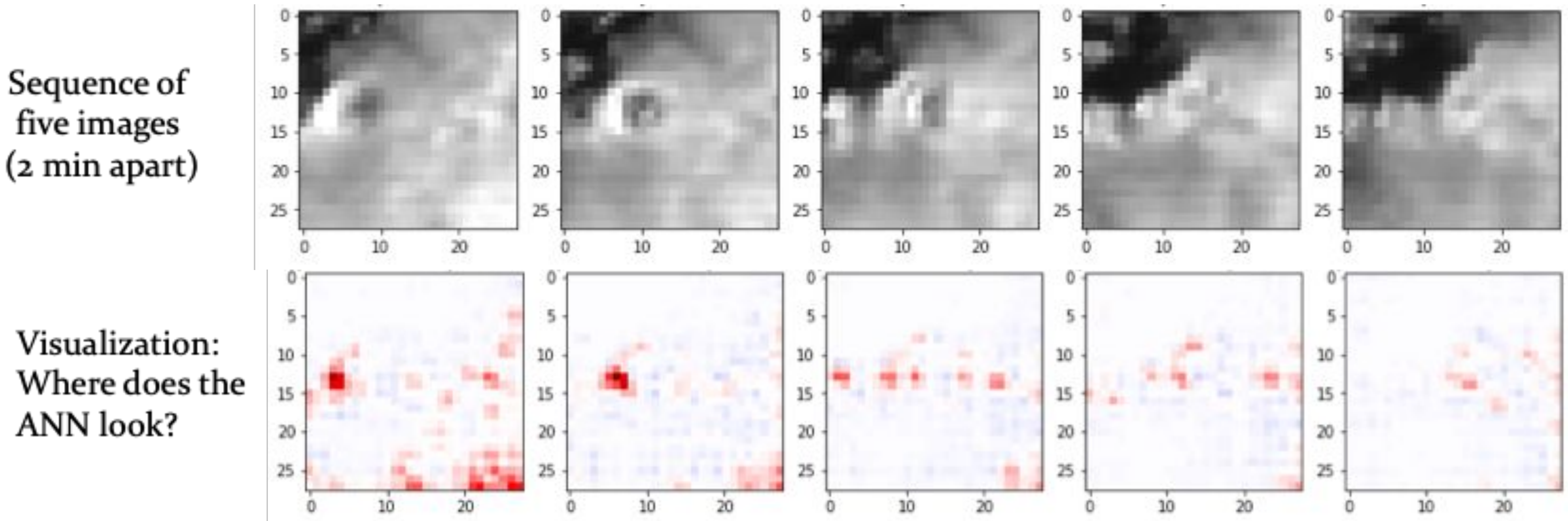
- innvestigate package
    - lots of options
    - not perfect, but a useful tool for our group



LRP for debugging and designing ANNs

Application 1

- **Yoonjin Lee (ATS/CIRA)**
- **Task: Detecting from series of GOES images whether there is convection or not.**
- Method: 3D Convolutional Neural Networks (CNNs). Uses: x,y, time as three axes.

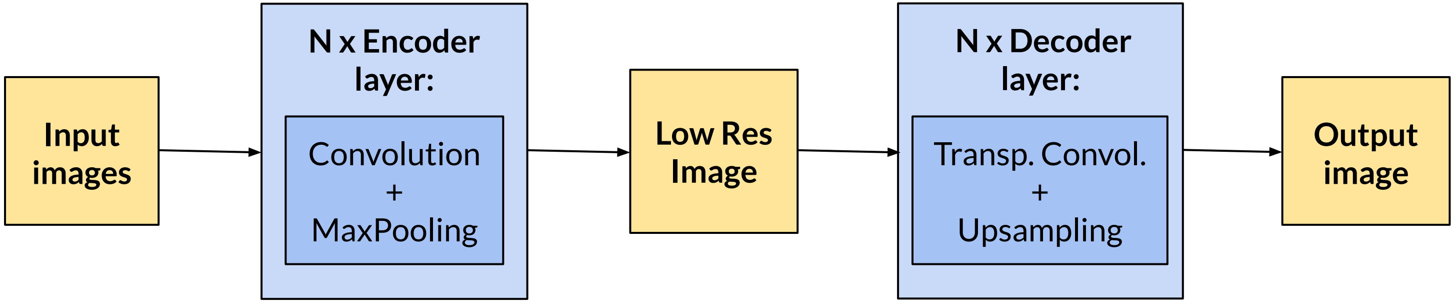


Desired strategy: ANN *should* look for combination of high brightness *and* bubbling.
Attribution map: Shows clear correlation to brightness, still exploring impact of bubbling. Work in progress.

Visualization → Brings it back to space of physics and expert knowledge! → Feedback for ANN design.

Application 2

- **Kyle Hilburn (CIRA)**
- Task: Generating synthetic radar images from GOES channels
- Input: Images from selected GOES channels.
- Output: Emulation of corresponding MRMS output.
- Method:
  - Convolutional Neural Networks (CNNs).
  - Encoder-decoder architecture that uses downscaling and upscaling.



Key question: How many encoder and decoder layers should we use? N = ?

Key concept: Receptive Field

Important concept for design of fully convolutional NNs

1) Theoretical Receptive Field (TRF):

- Definition: Size of input area that can - theoretically - affect a single output pixel.
- Easy to calculate for CNN, depends only on architecture.
- But in practice: only small subset (central region) truly comes into play.

2) Effective receptive field (ERF):

- Definition: Size of input area that practically affects a single output pixel.
- ERF generally much smaller than TRF.
- Not easy to calculate - ERF even changes throughout network training.
- ERF is important for network design: max feature size that CNN can recognize.

Great discussion of effective receptive fields:

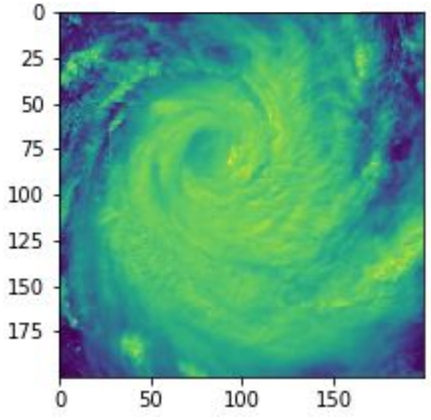
Luo, Wenjie, et al. “Understanding the Effective Receptive Field in Deep Convolutional Neural Networks.” *Advances in Neural Information Processing Systems*, papers.nips.cc, 2016, pp. 4898–906, http://papers.nips.cc/paper/6202-understanding-the-effective-receptive-field-in-deep-convolutional-neural-networks.

What does LRP give us?

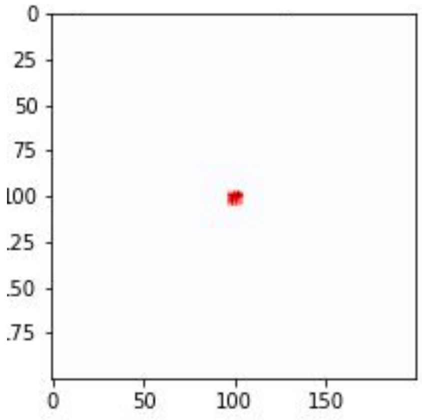
Shown below are LRP results

- For a single output pixel (here chosen at center of image).
- For network with varying architecture - N *encoder layers* and N *decoder layers* (N=1,2,3).

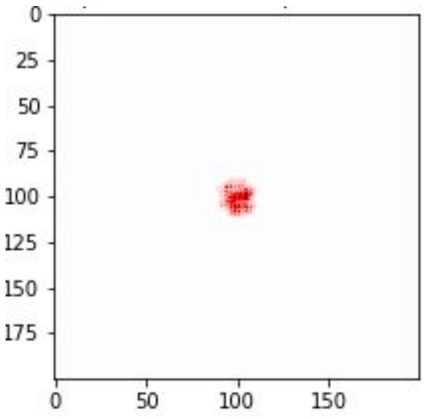
Input Image



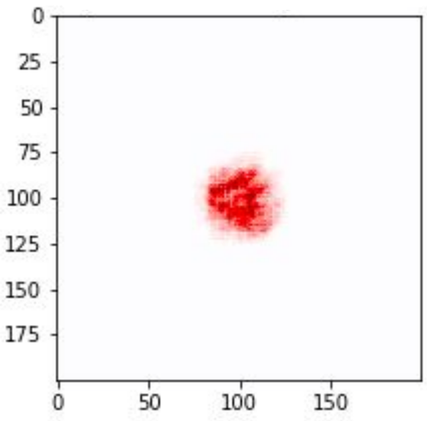
LRP for N=1



LRP for N=2



LRP for N=3



Result: **LRP gives good estimate of "effective receptive field"**

→ Found simple way to approximate ERF!

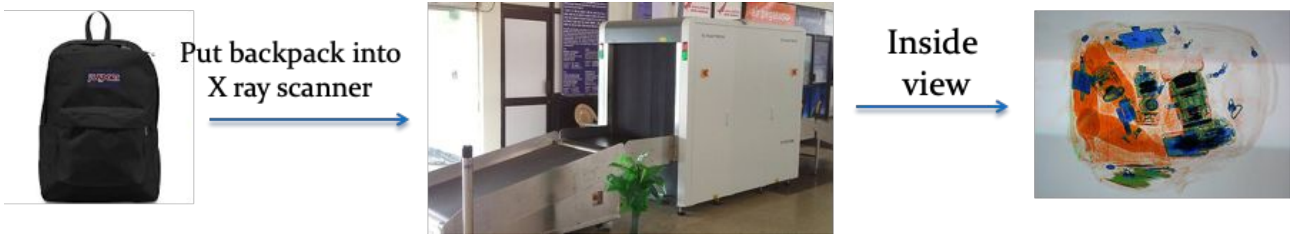
→ Tells us patch size of input image being considered for each output pixel.

→ Match patch size with size of meteorological feature you want network to use as context.

Some Limitations

- Many tools are developed primarily for classification tasks.
  → Using them for regression tasks → “Off-label use”. Work-arounds for now. Work in progress.

- Heat maps only give you <u>locations</u> where ANN is looking, but sometimes that’s not enough to completely figure out strategy. Recall analogy below.



Put backpack into X ray scanner

Inside view

- Keep it simple - an ANN with fewer layers is easier to understand. Like a backpack with fewer “layers”.

- Heat maps are specific for each input. May have to look through many cases (or take averages, or perform clustering, etc.).

LRP for scientific discovery

What we mean by “scientific discovery”

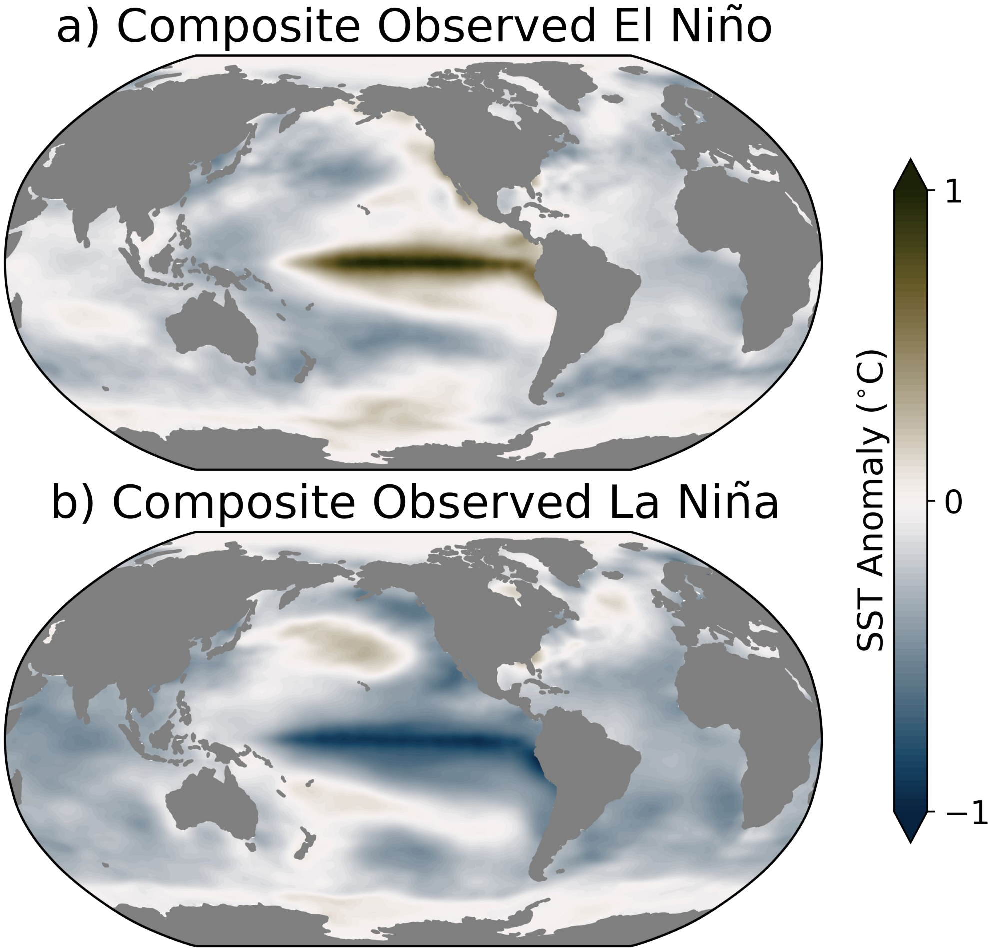
- Many neural network tasks focus on maximizing the accuracy of the neural network
- We emphasize designing a neural network to maximize the amount of scientific value that can be offered by the interpretation
    - Combination of prediction accuracy, appropriate structure for interpretation methods, etc.
    - A Goldilocks problem!

ENSO

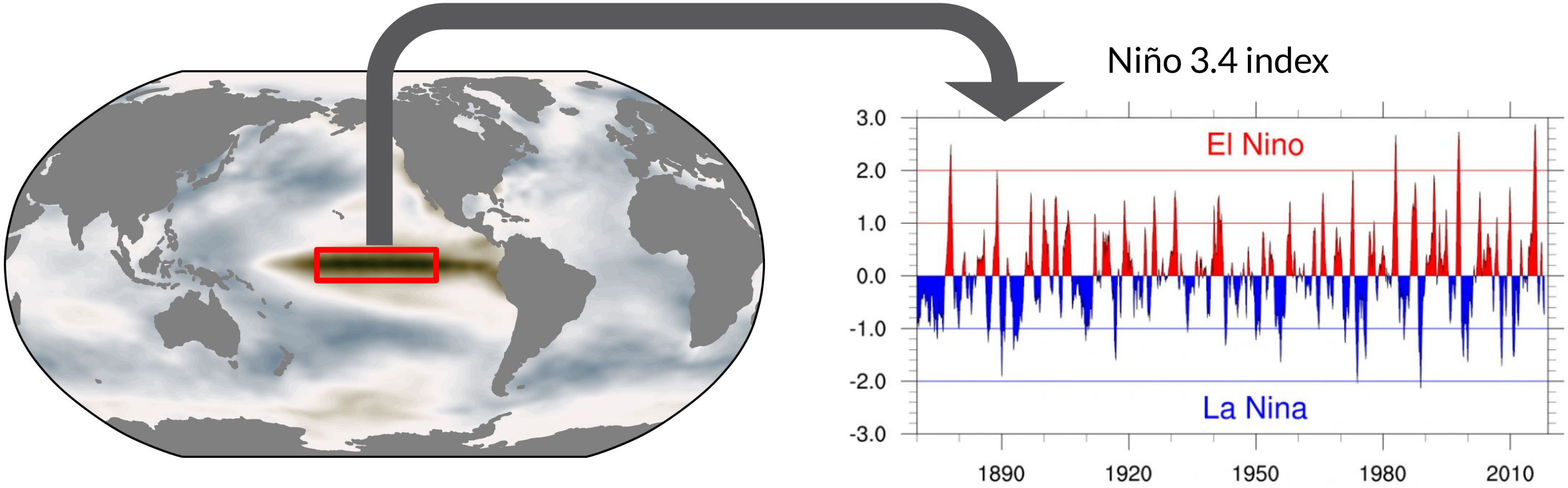
Starting with an example we know the answer to.

What is ENSO?

- Dominant mode of sea-surface temperature variability within the tropical Pacific
- Impacts weather and climate across the globe



Defining ENSO…



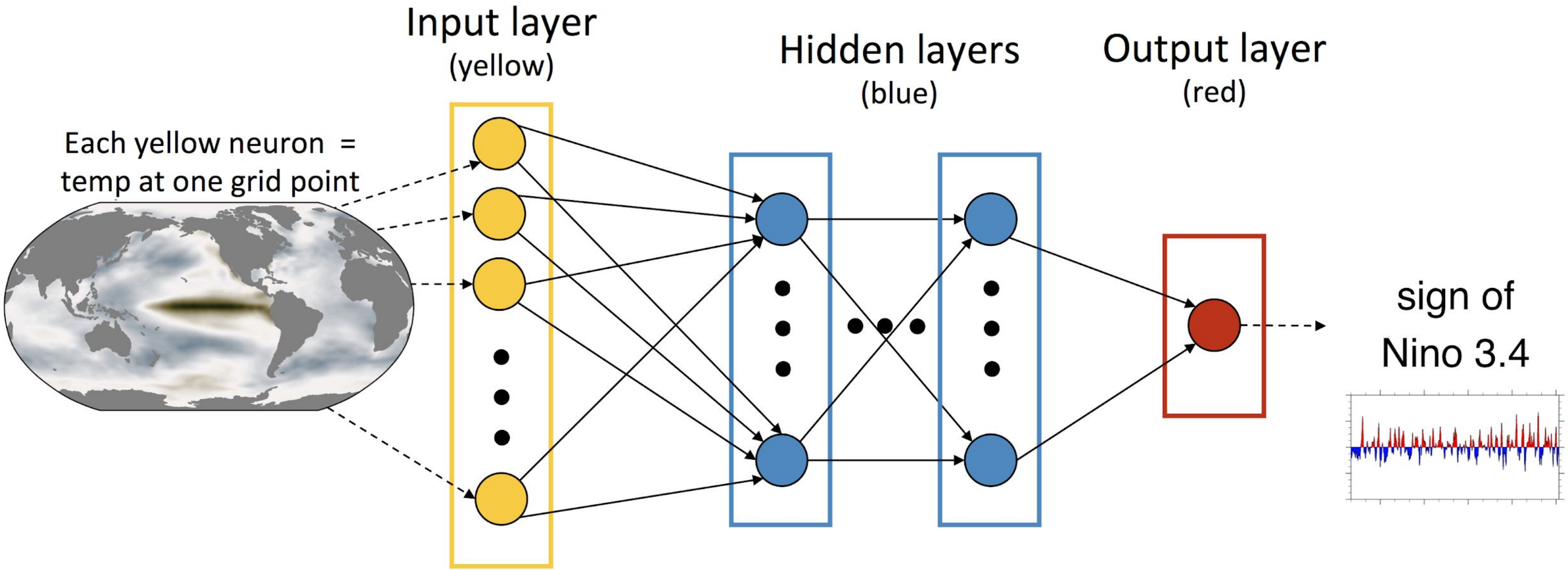
Niño 3.4 index

Figure courtesy of NCAR Climate Data Guide

ENSO is commonly defined according to average sea-surface temperatures within the central tropical Pacific.

Toms et al. (in prep)

ENSO + Neural Networks



Colorado State University

Toms et al. (in prep)

What we're testing with this example...

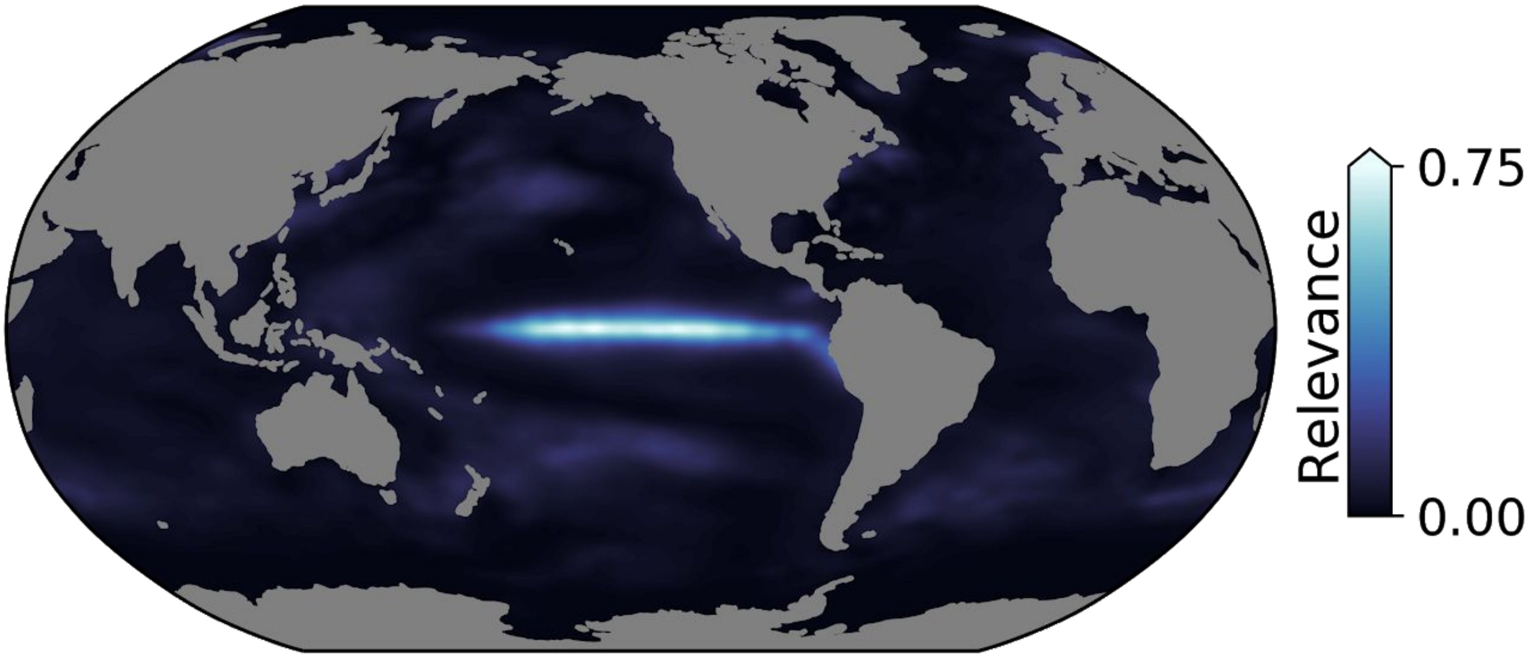
The network classifies the sign of ENSO with 100% accuracy, so...

- Is the neural network interpretation physical? Does it depict ENSO spatial patterns consistent with physical theory?

- Assume we don't know the answer: does the neural network focus only on the Nino3.4 region, or does it help tell us about the full spatial pattern of ENSO?

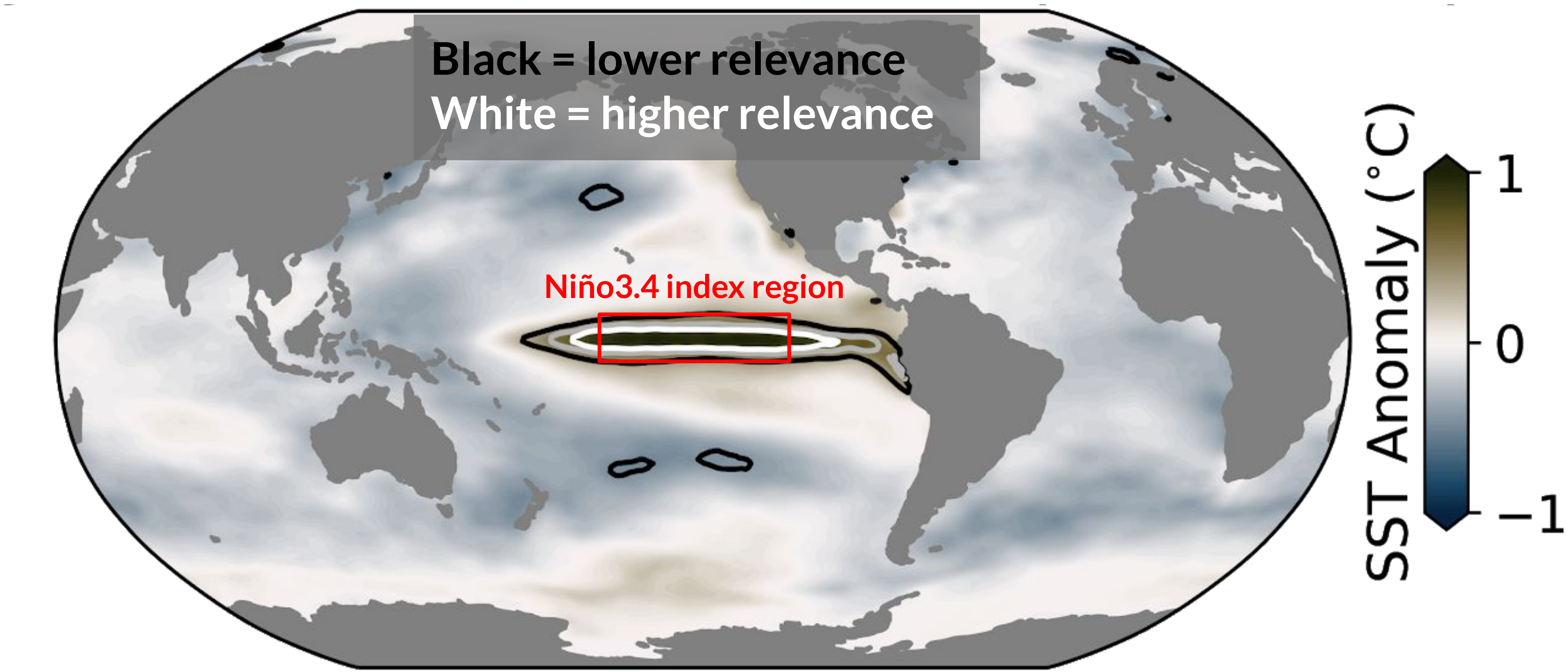
Toms et al. (in prep)

Composite relevance for all El Niño samples



Toms et al. (in prep)

Relevance overlays robust El Niño signal



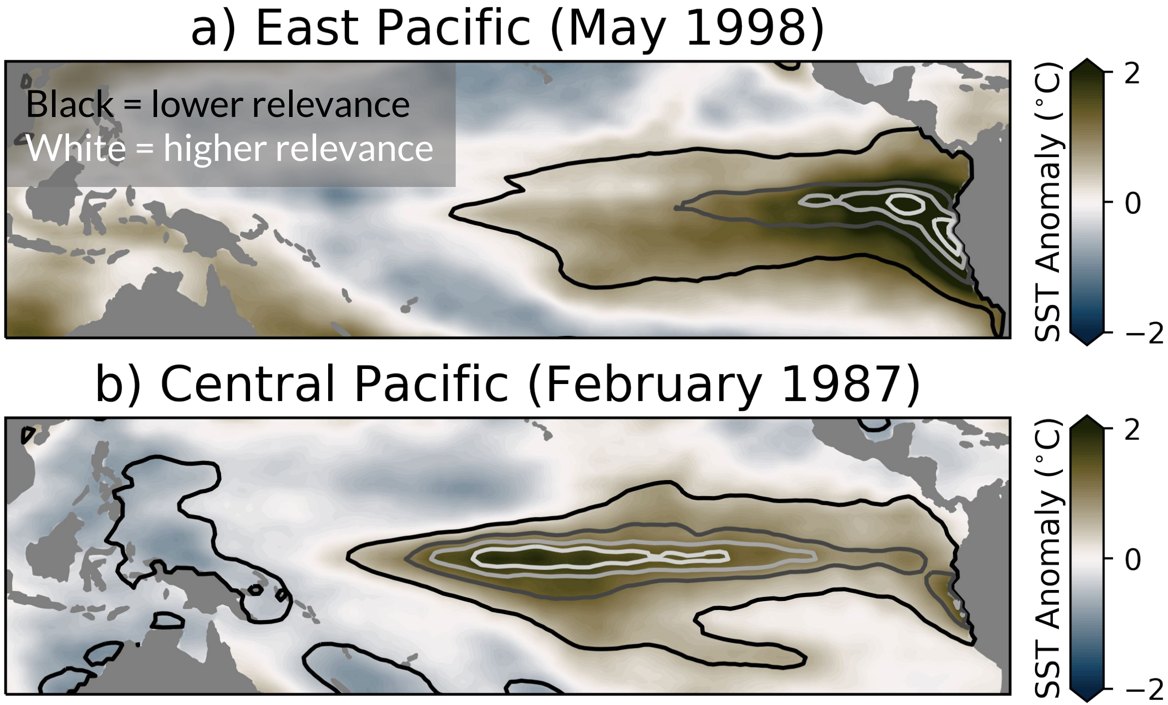
Black = lower relevance
White = higher relevance

Niño3.4 index region

Toms et al. (in prep)

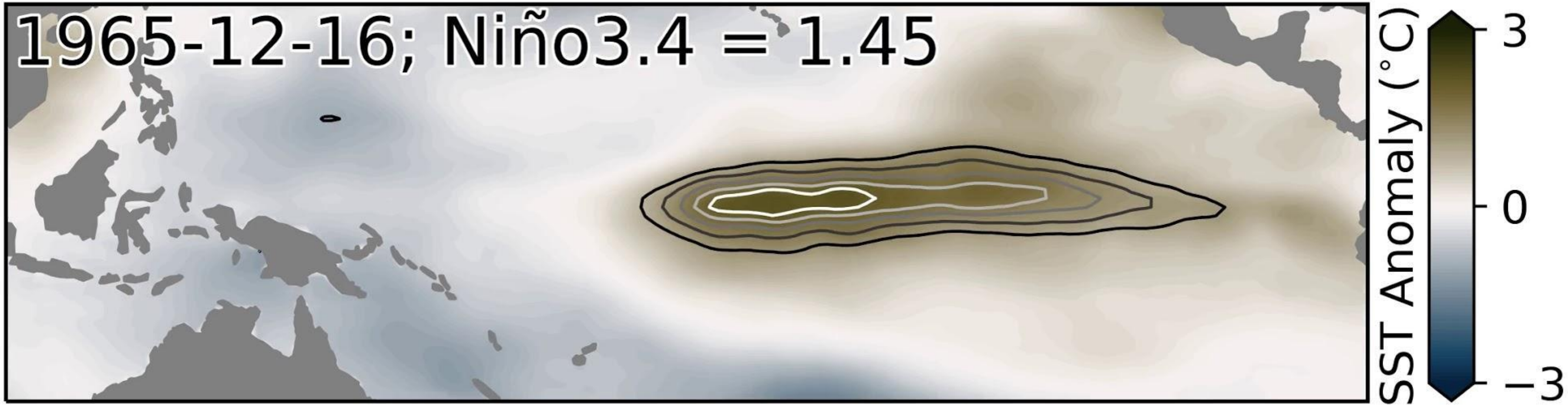
Capturing different modes of variability

- The neural network can identify different types of El Nino events
- Attention is refocused depending on the location of sea-surface temperature anomalies
- **If not known a-priori, LRP could show that multiple modes of ENSO exist**



Toms et al. (in prep)

Animated relevance for ENSO…



LRP highlights where the neural network focuses its attention for each sample and shows which patterns are the most relevant for the neural network's decision

Toms et al. (in prep)

We have shown that LRP can be used to identify known patterns of climate variability.

Now we extend this idea to unknown patterns of climate variability...

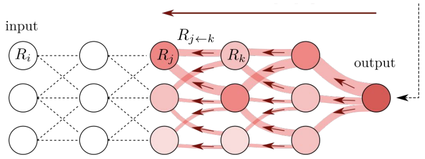
Identifying Forced Patterns of Change

within CMIP5 models and observations

Visualization tools for scientific discovery

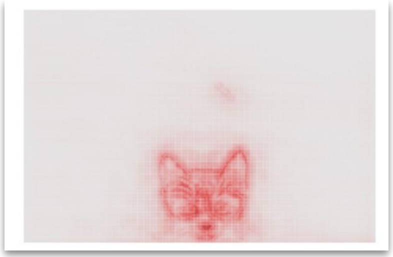
- ANNs for science need not be only focused on prediction - prediction can be a means to an end
- Attribution methods like LRP allow us to frame the problem such that **what the ANN learns** is the science
- Get creative!

Wrap-up



Limitations of LRP

- Initially developed for classification using ReLu
- Unclear how to interpret the output for regression
- Documentation is not great, and different publications seem to conflict with each other
- Must consider network architecture for best visualization ahead of time (e.g. avoid max-pooling)



Parting thoughts

- Artificial neural networks are **not black boxes anymore**
- Many visualization tools out there - let’s use them!
- They can be used for debugging, design and science
- While many tools are coming out of computer science every week, they are not optimized for our applications: more collaboration needed

Resources

Software *(we have not used all of these ourselves)*:

- heatmapping.org website: http://heatmapping.org/
- innvestigate toolbox [Keras]: https://github.com/albermax/innvestigate
- Keras-vis package for Keras: https://github.com/raghakot/keras-vis
- LRP pytorch package: https://github.com/moboehle/Pytorch-LRP

Further reading:

- **Intro to Feature Visualization:** Olah, C., et al. “Feature Visualization.” Distill, 2017, https://distill.pub/2017/feature-visualization/.
- **Book on Explainable AI:** W Samek, G Montavon, A Vedaldi, LK Hansen, KR Müller (Eds.) Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, Springer LNCS 11700, 2019

The End