CS273B: Deep learning for Genomics and Biomedicine

Lecture 9: Methods for interpreting neural nets

10/22/2017

Anshul Kundaje, James Zou

(Slides contributed by Av Shrikumar)

1

Perturbation methods

2

Backprop methods

3

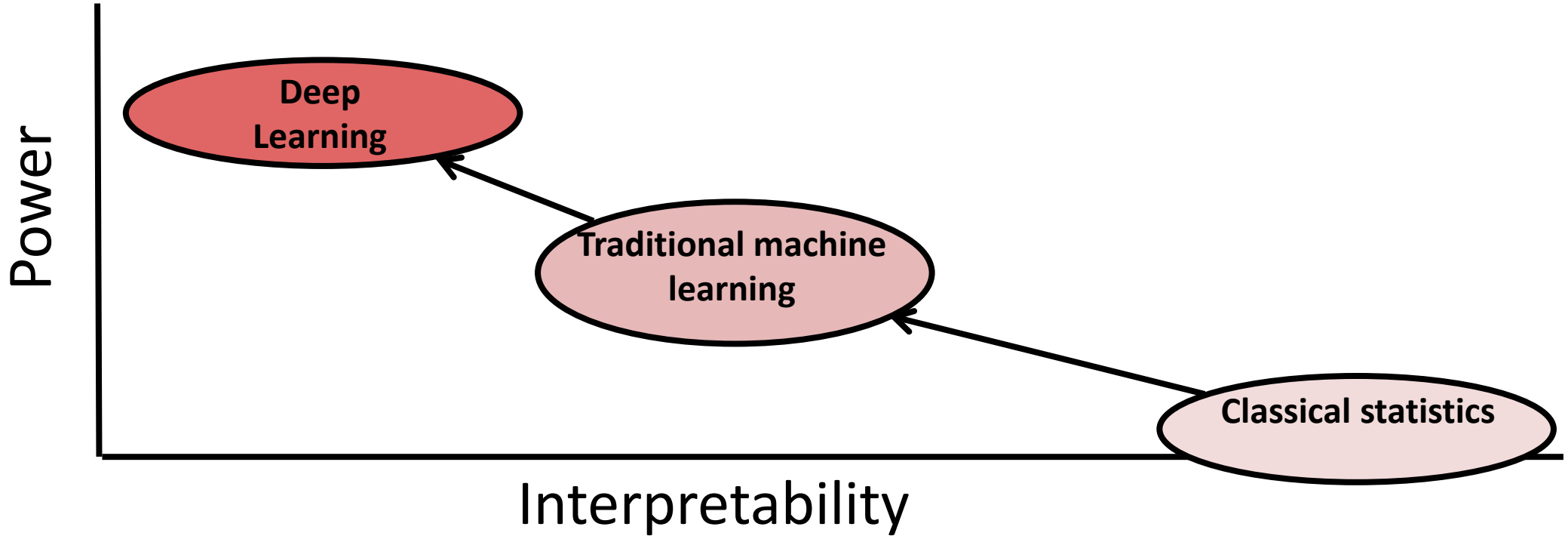
Aggregation methods

With great power comes really poor interpretability…

NATURE | NEWS FEATURE

Can we open the black box of AI?

Artificial intelligence is everywhere. But before scientists trust it, they first need to understand how machines learn.

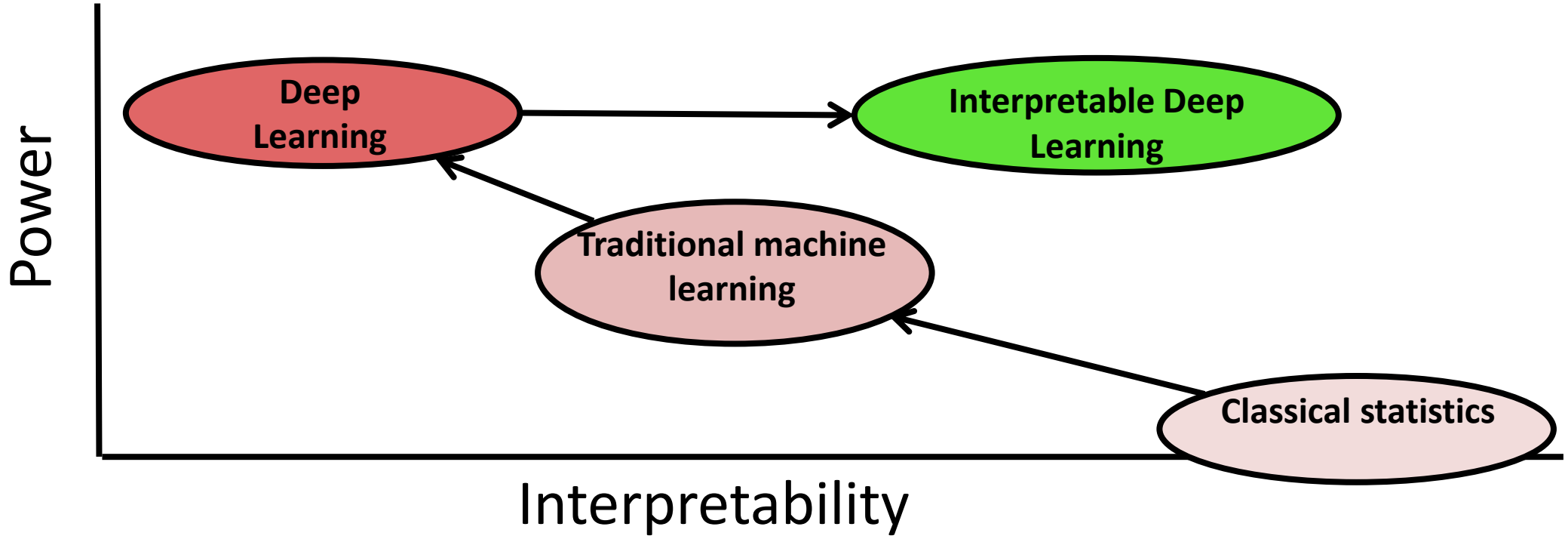


With great power comes really poor interpretability…

NATURE | NEWS FEATURE

Can we open the black box of AI?

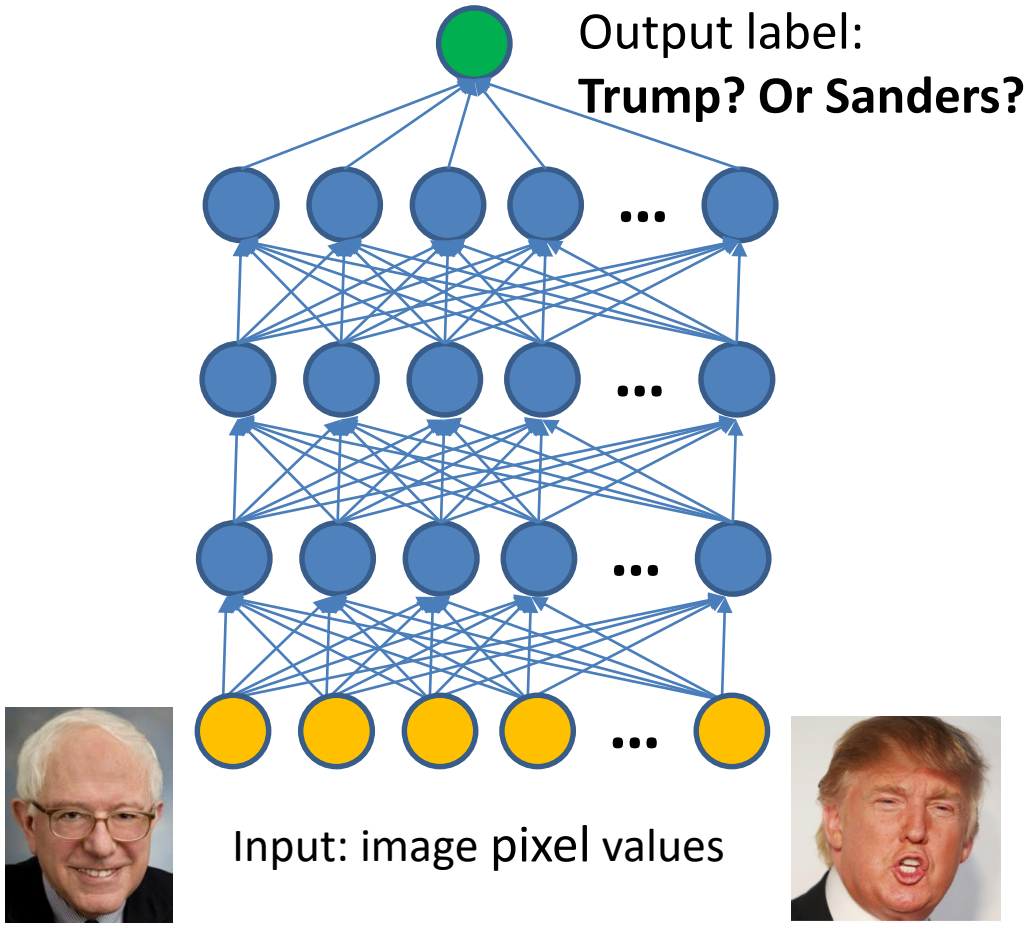
Artificial intelligence is everywhere. But before scientists trust it, they first need to understand how machines learn.



Identifying important features in individual input examples

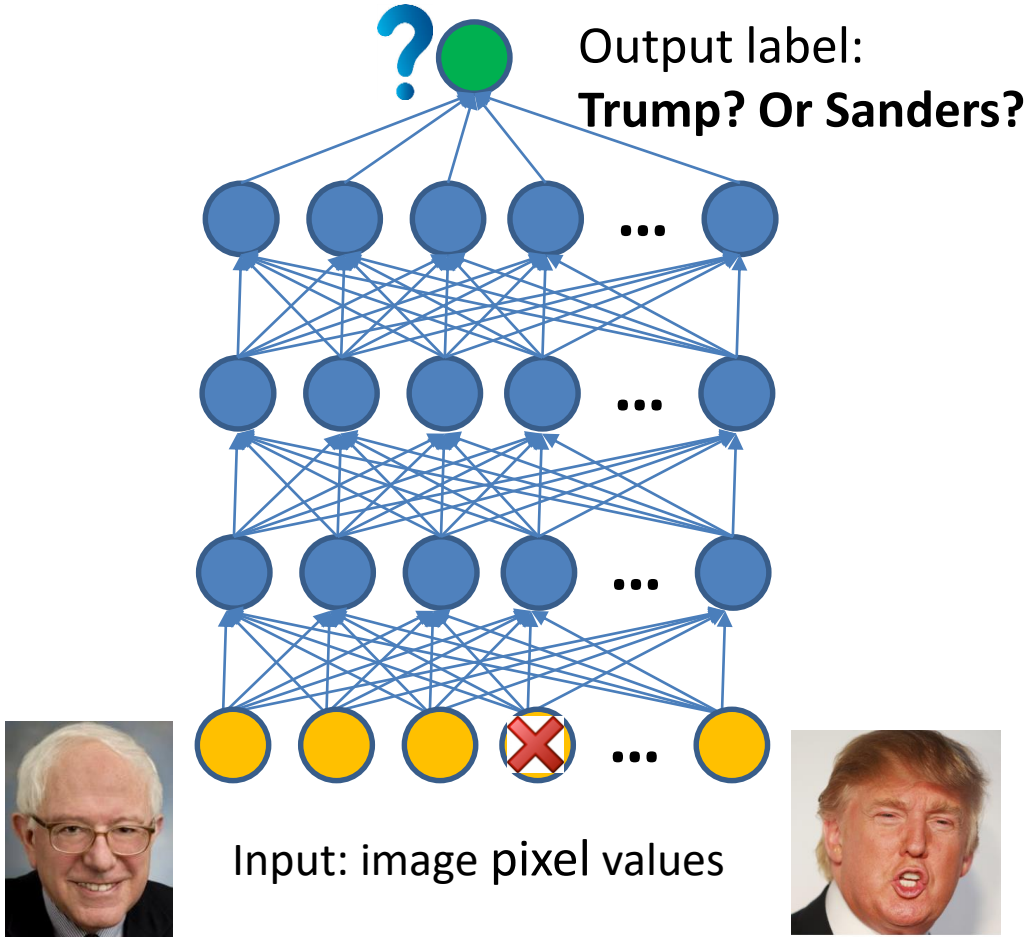
How can we compute importance scores?

Idea 1: perturbation



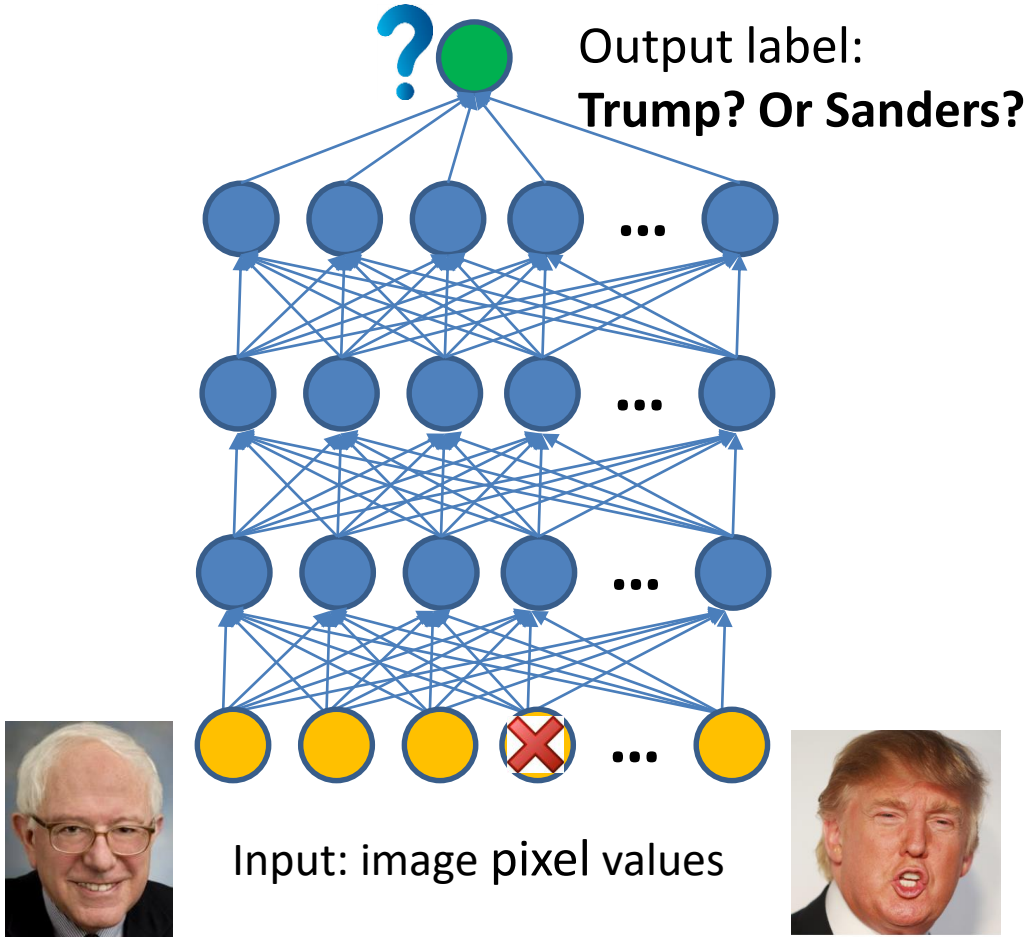
How can we compute importance scores?

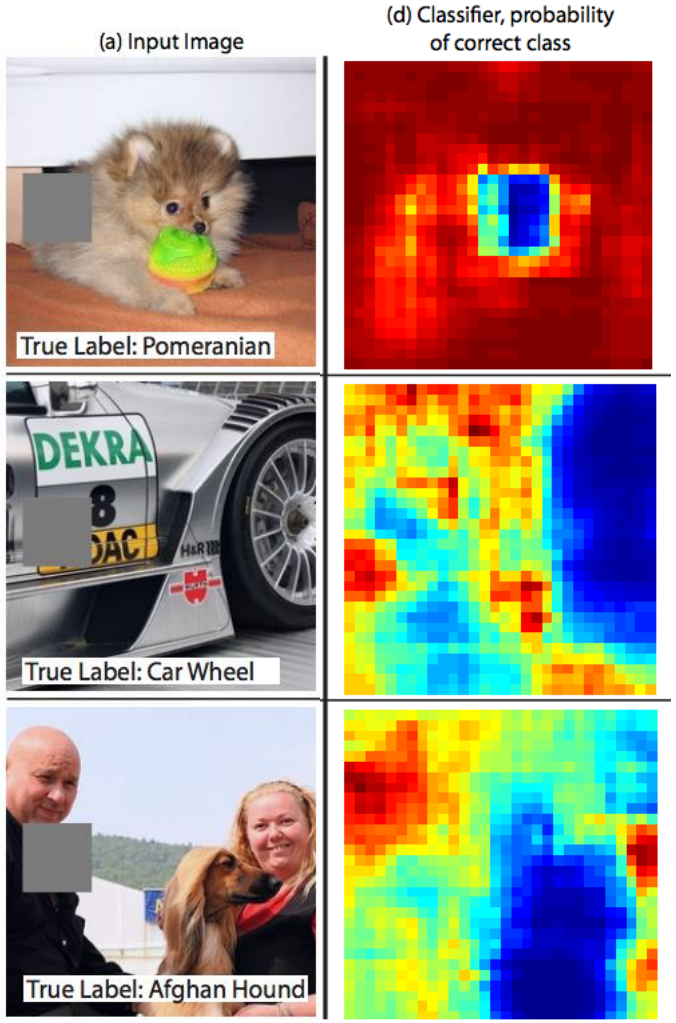
Idea 1: perturbation



How can we compute importance scores?

Idea 1: perturbation

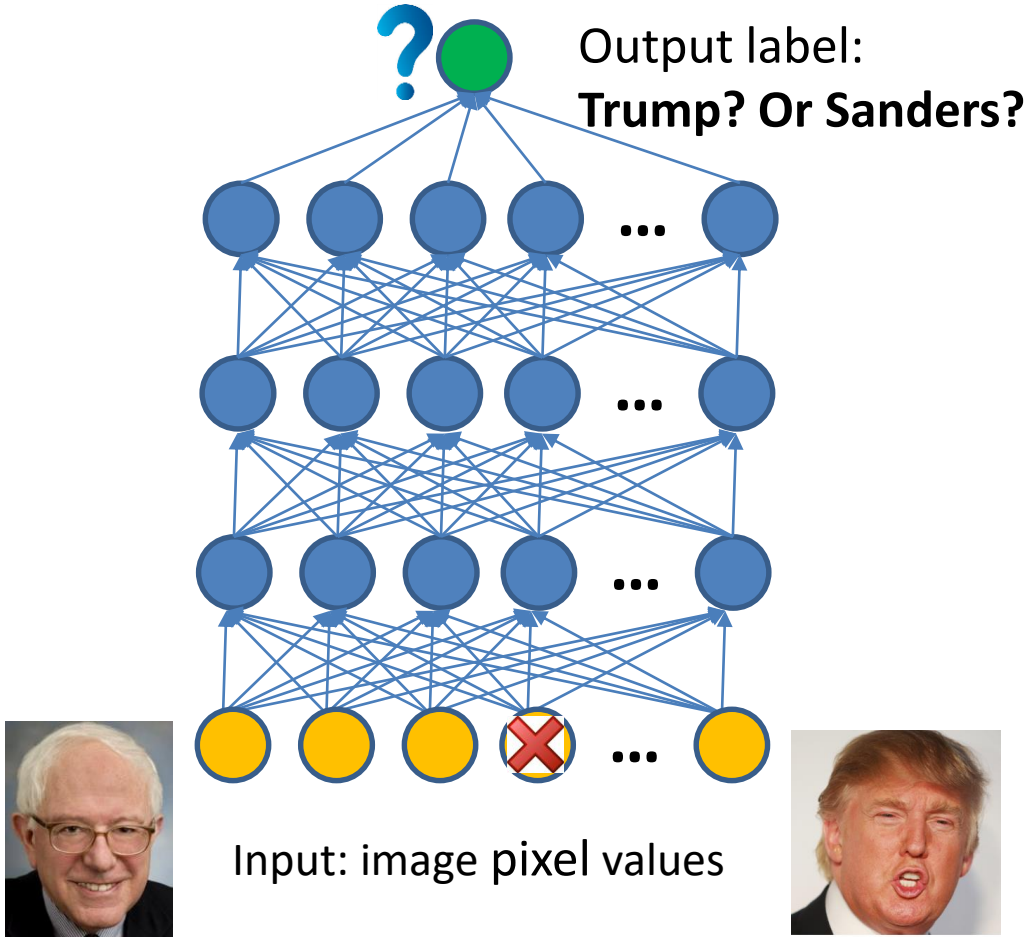


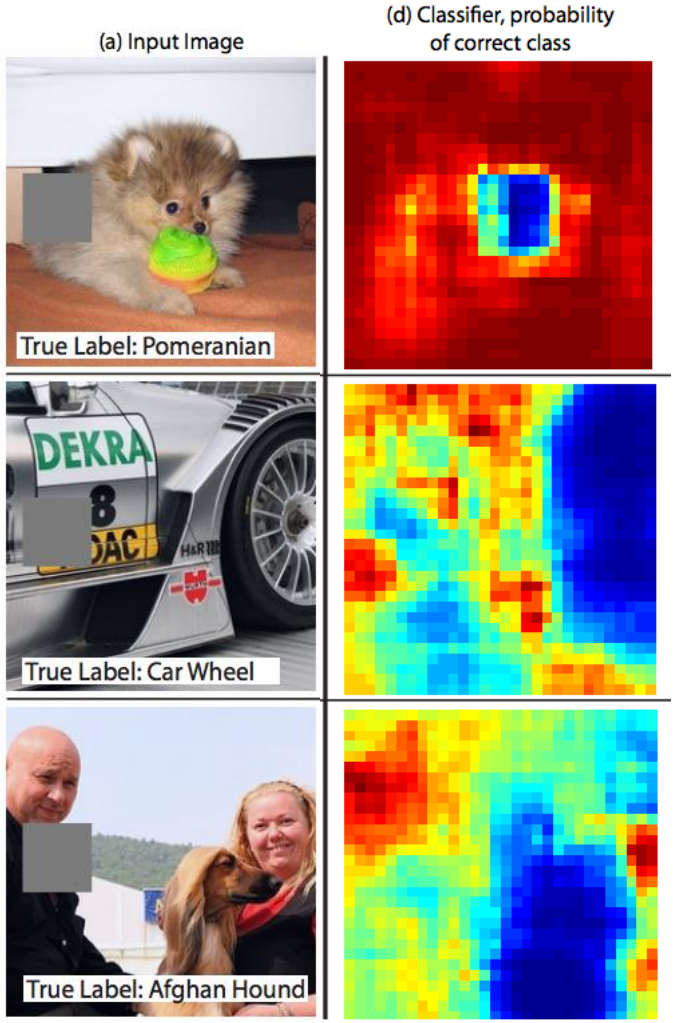


https://arxiv.org/pdf/1311.2901v3.pdf

How can we compute importance scores?

Idea 1: perturbation

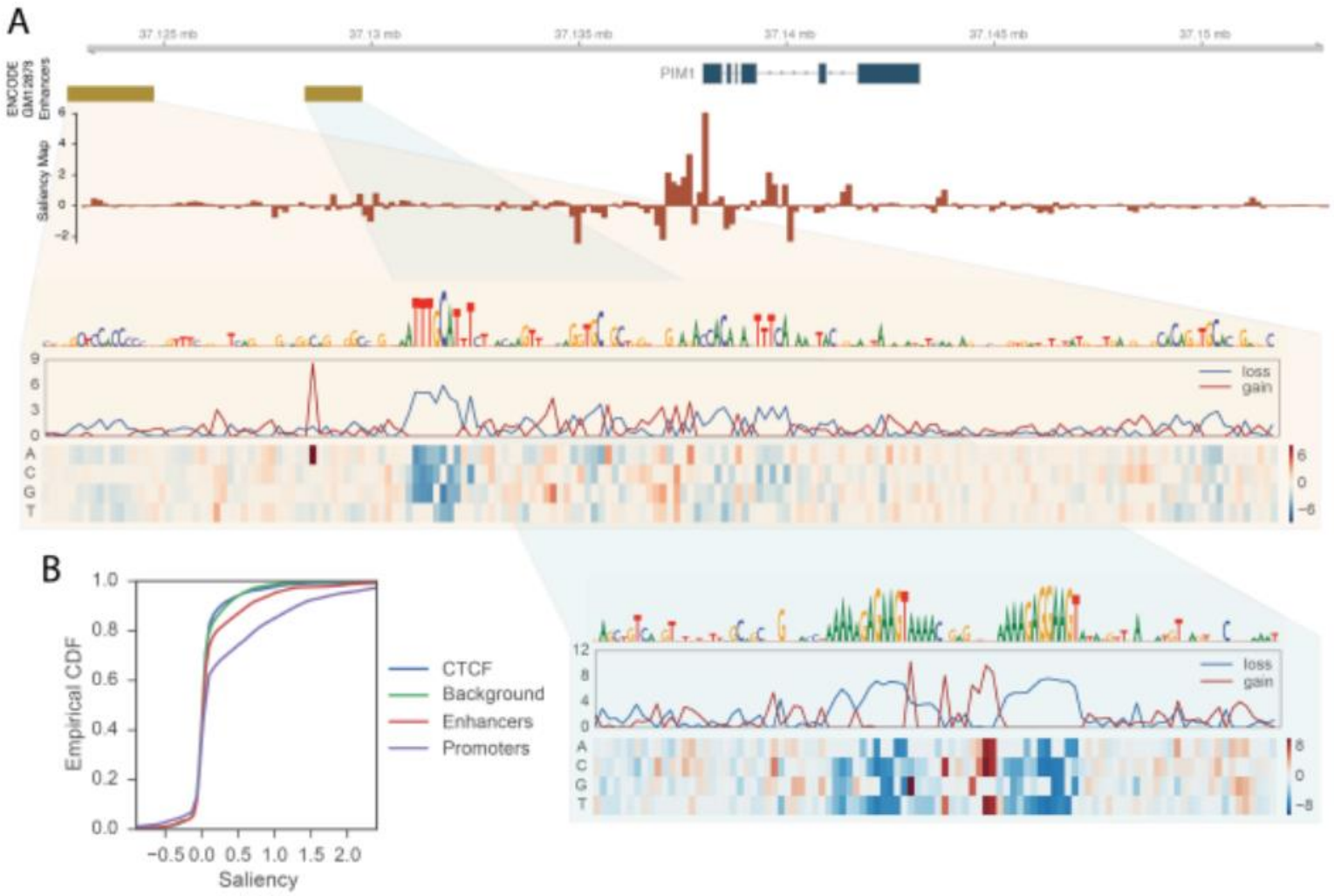




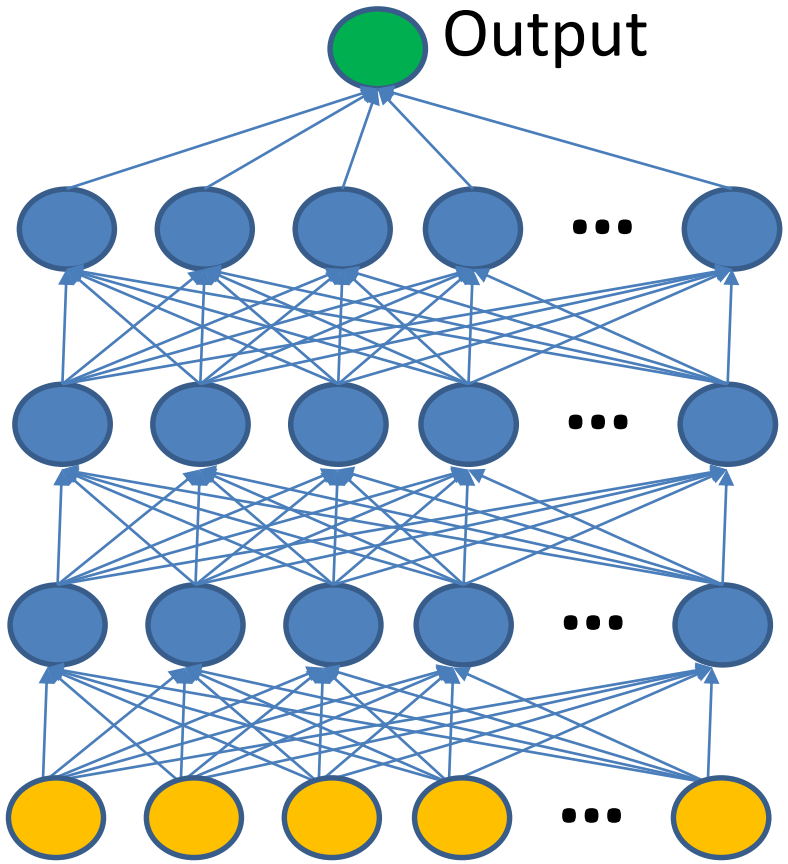
https://arxiv.org/pdf/1311.2901v3.pdf

"In-silico mutagenesis" in the context of DNA sequence

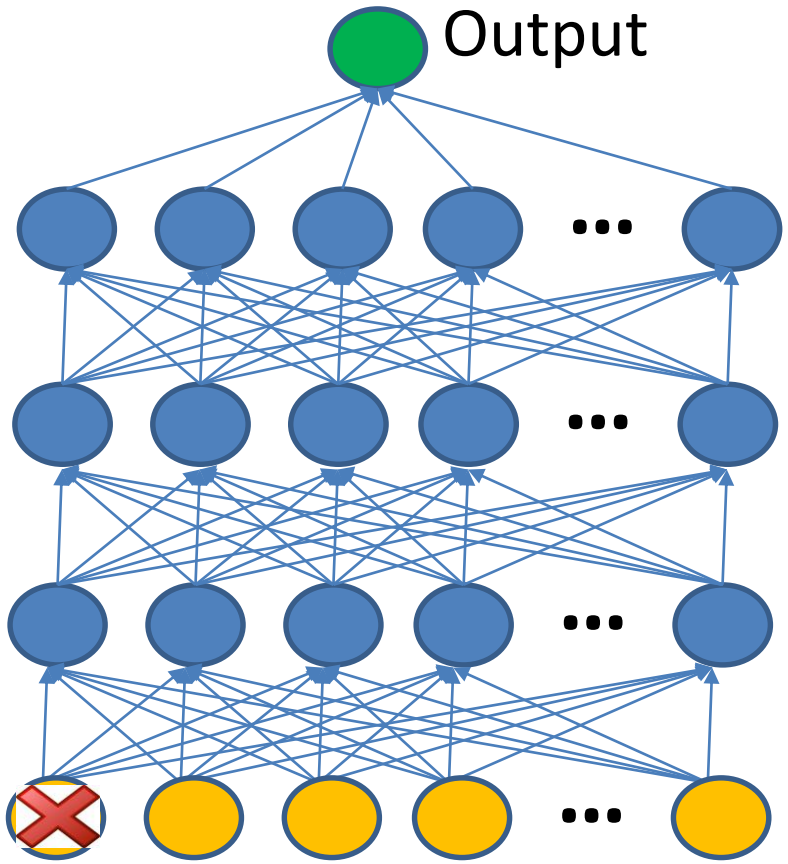
In-silico mutagenesis in the Basenji model to identify important nucleotides



Drawbacks of perturbation approach

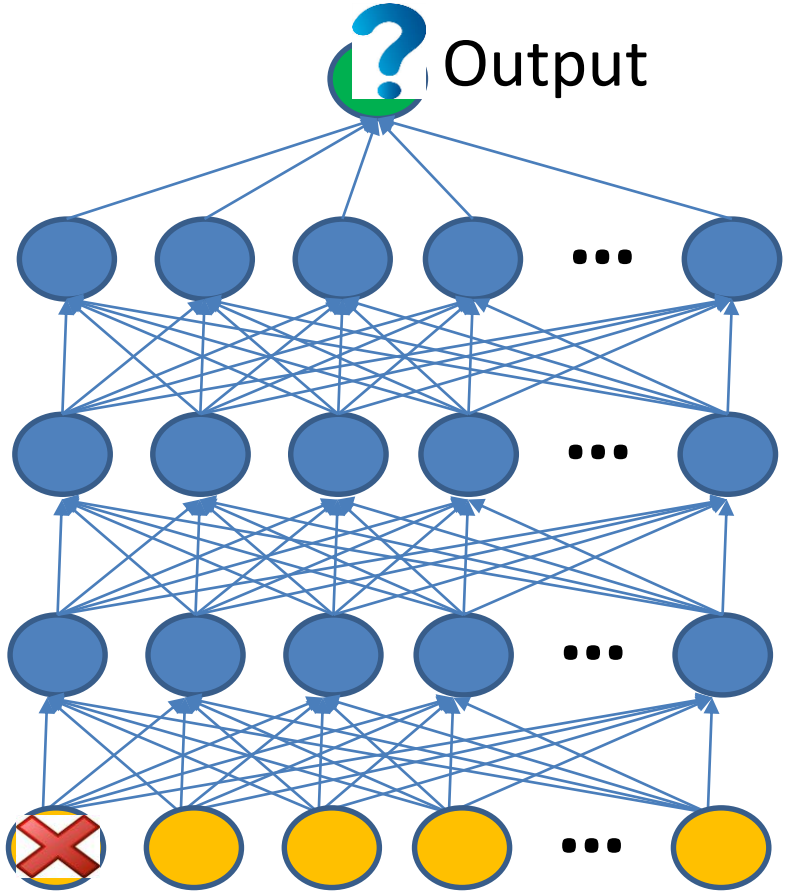


Drawbacks of perturbation approach

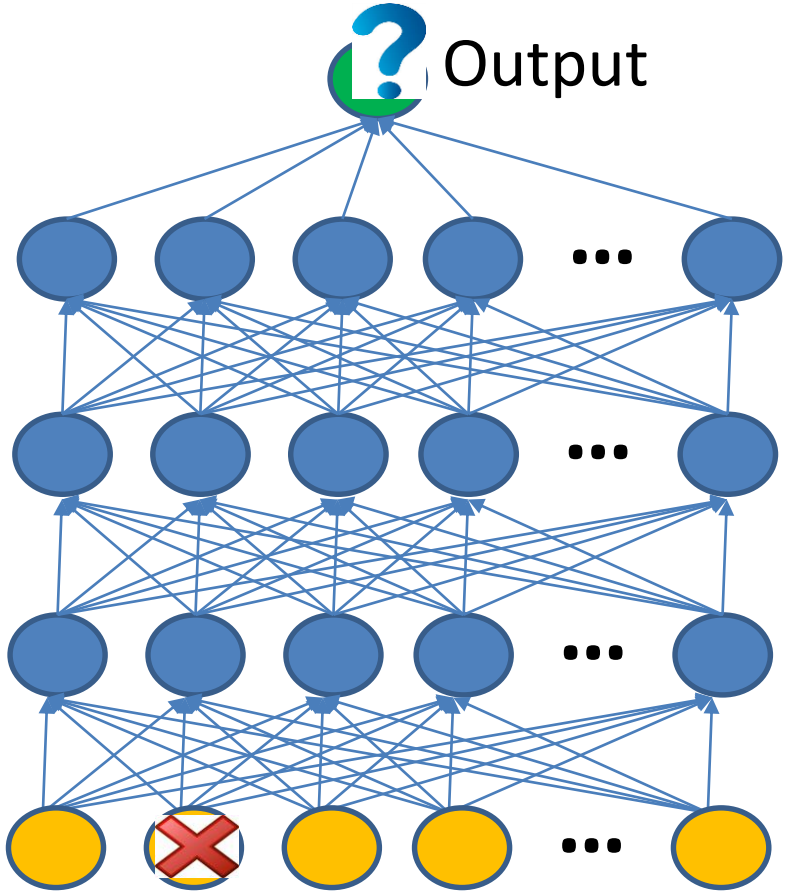


Yellow = inputs

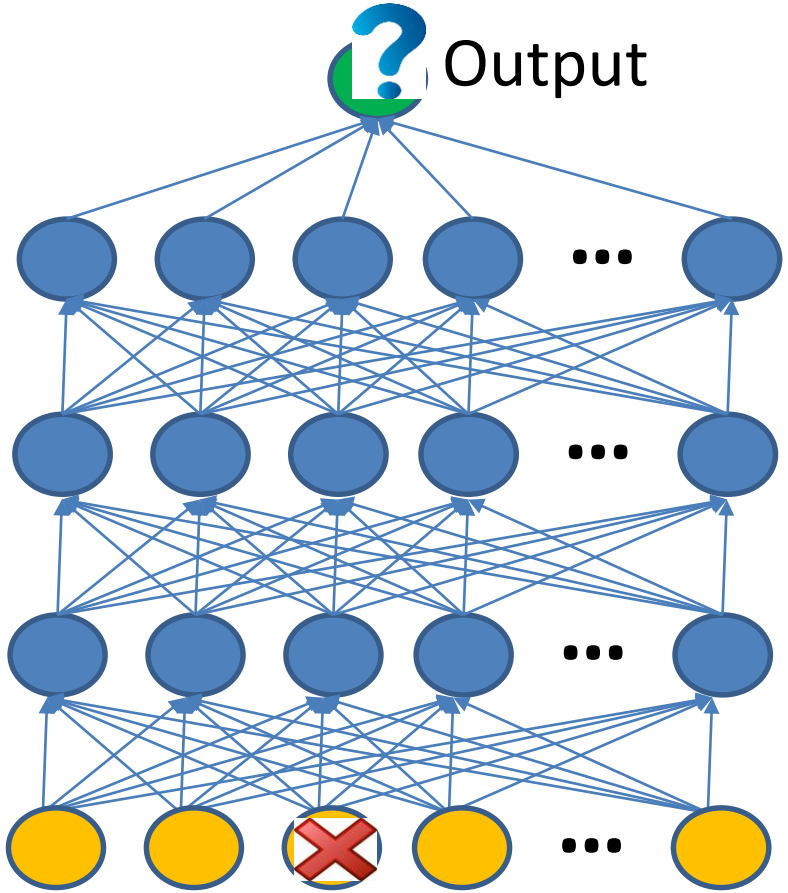
Drawbacks of perturbation approach



Drawbacks of perturbation approach

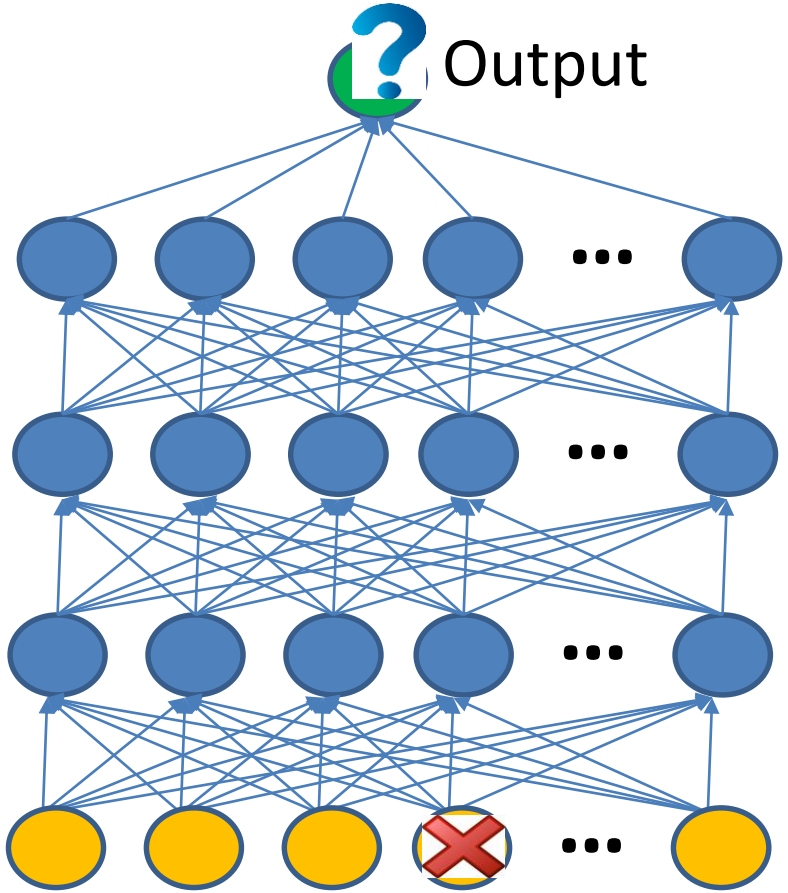


Drawbacks of perturbation approach



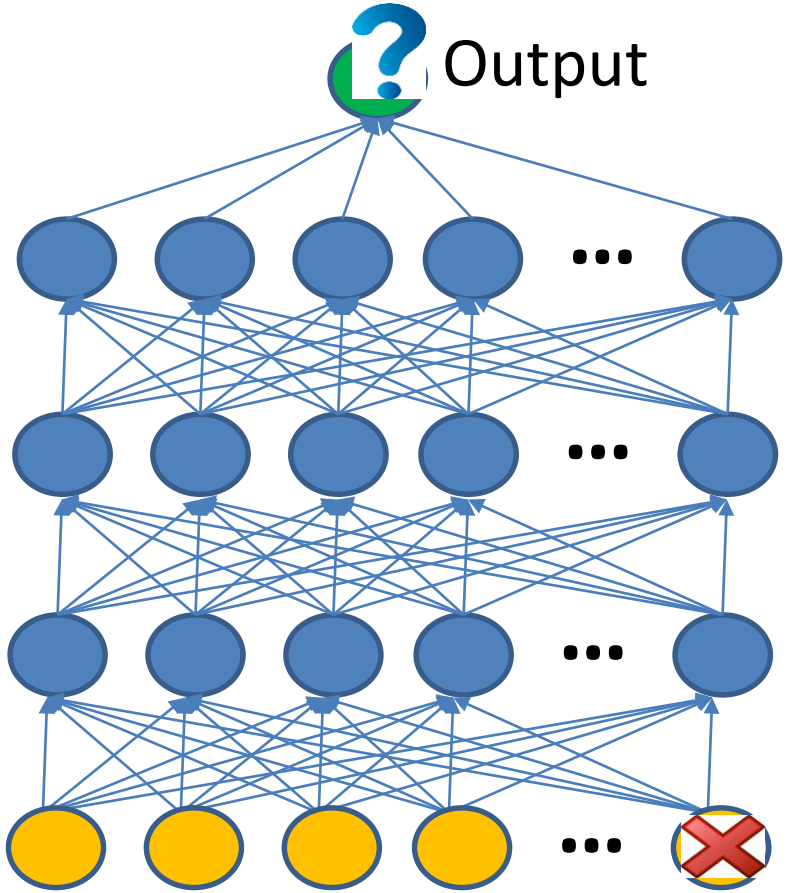
Yellow = inputs

Drawbacks of perturbation approach



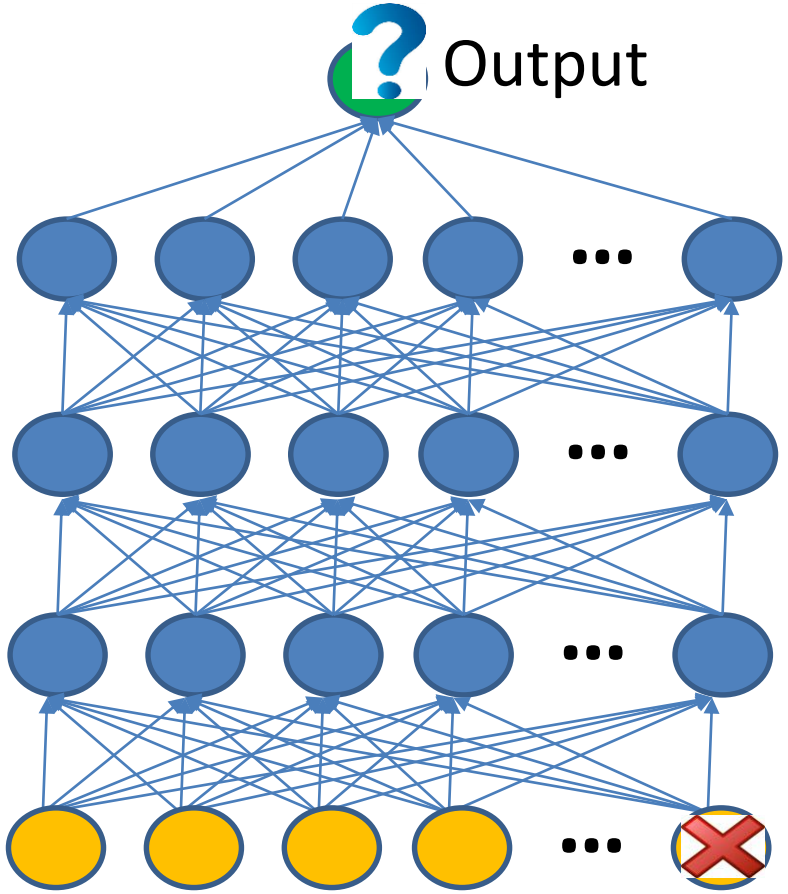
Yellow = inputs

Drawbacks of perturbation approach



Yellow = inputs

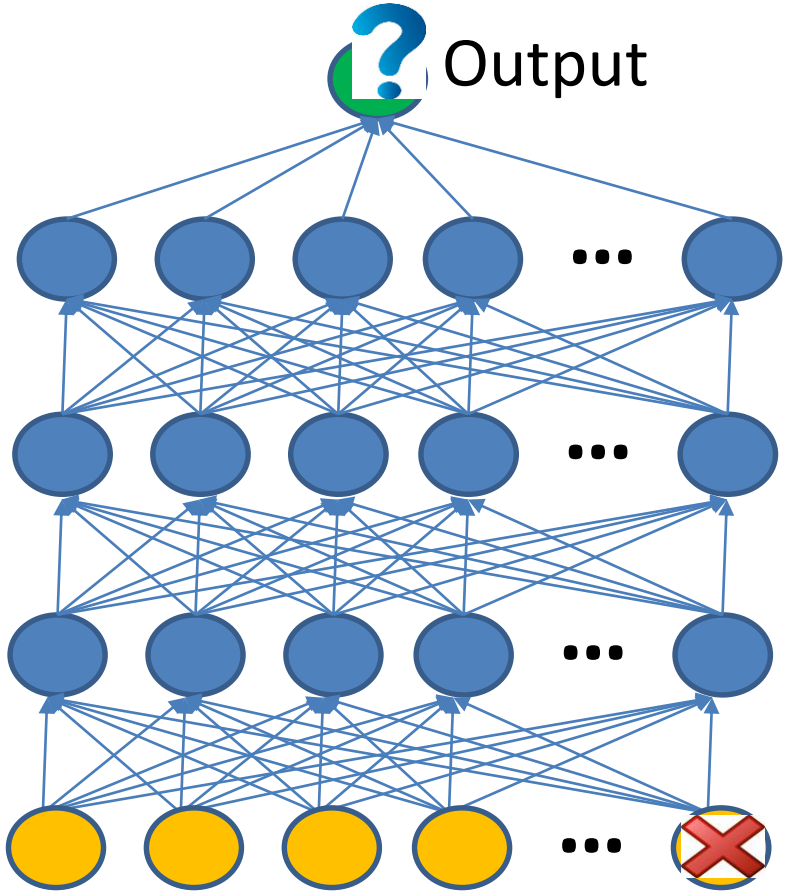
Drawbacks of perturbation approach



Examples

1) Zeiler & Fergus, 2013
2) LIME (Ribeiro et al., 2016)
3) Zintgraf et al., 2017

Drawbacks of perturbation approach



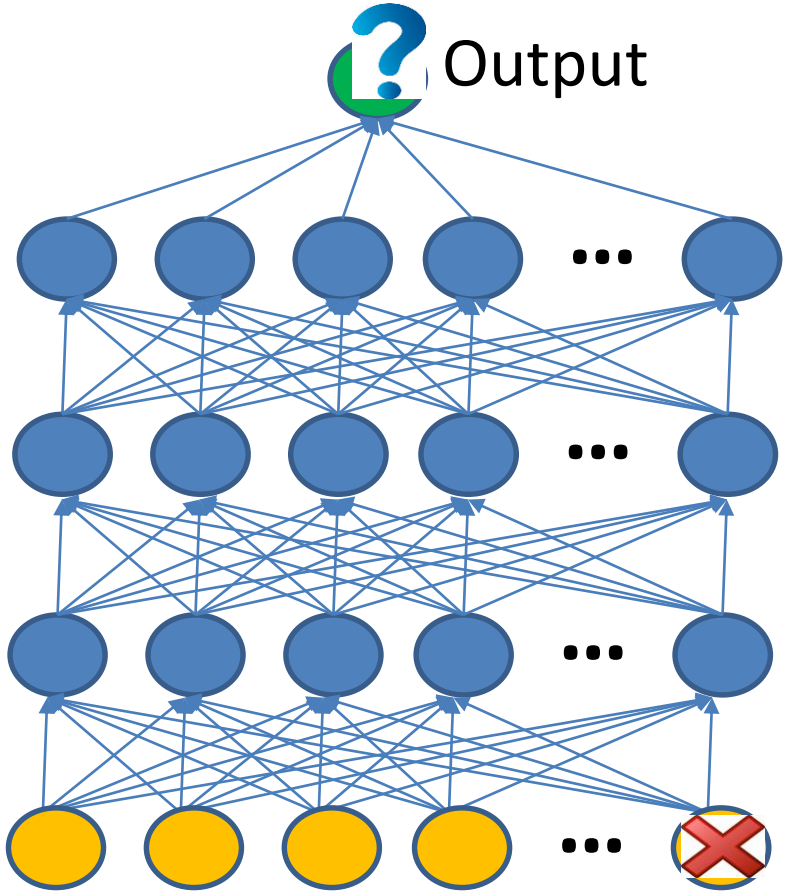
Examples

1) Zeiler & Fergus, 2013
2) LIME (Ribeiro et al., 2016)
3) Zintgraf et al., 2017

Drawbacks

1) Computational efficiency - requires one forward prop for each perturbation

Drawbacks of perturbation approach



Examples

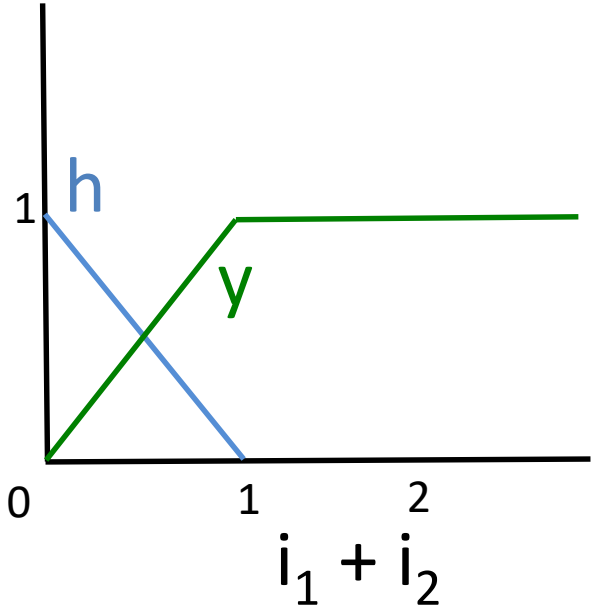
1) Zeiler & Fergus, 2013
2) LIME (Ribeiro et al., 2016)
3) Zintgraf et al., 2017

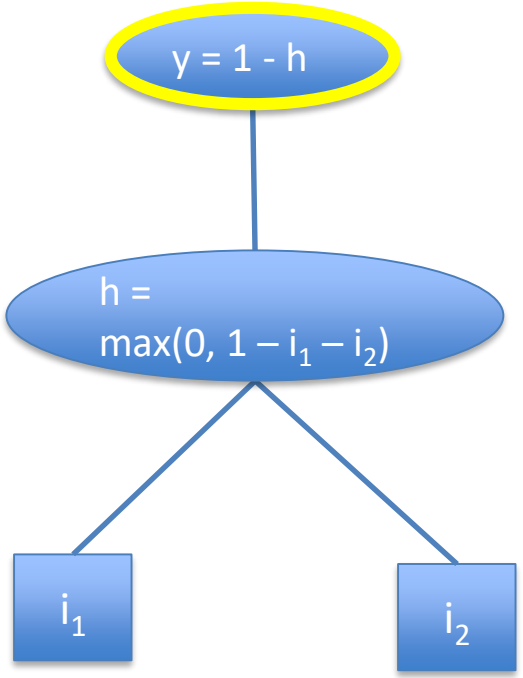
Drawbacks

1) Computational efficiency - requires one forward prop for each perturbation
2) Saturation

Saturation problem illustrated

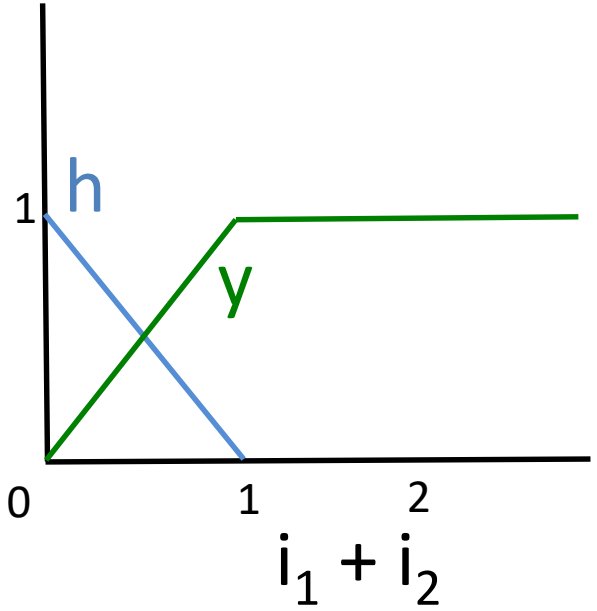
$$y = (i_1 + i_2) \text{ when } (i_1 + i_2) < 1$$
$$= 1 \quad \text{when } (i_1 + i_2) >= 1$$

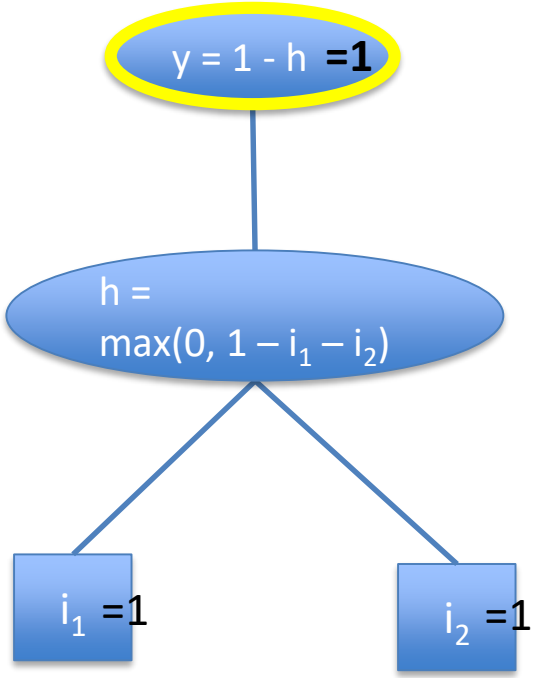




Saturation problem illustrated

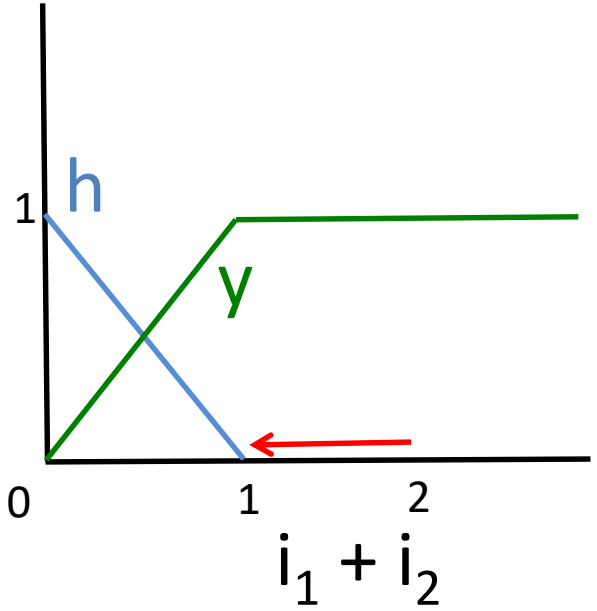
$y = (i_1 + i_2)$ when $(i_1 + i_2) < 1$
$= 1$ when $(i_1 + i_2) >= 1$

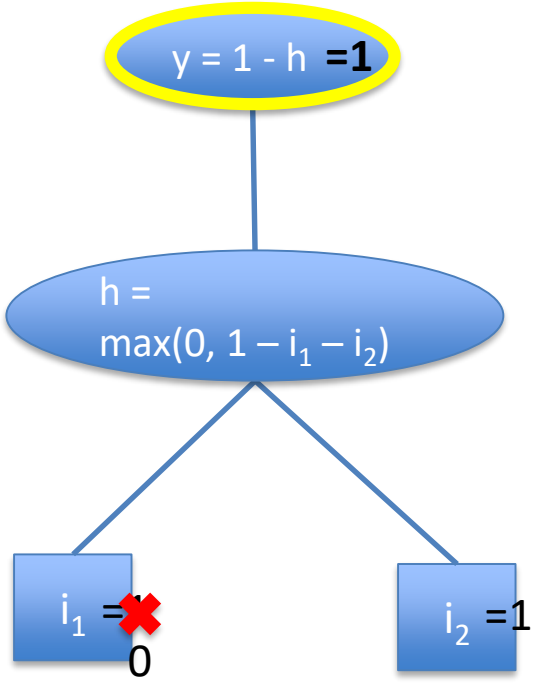




Saturation problem illustrated

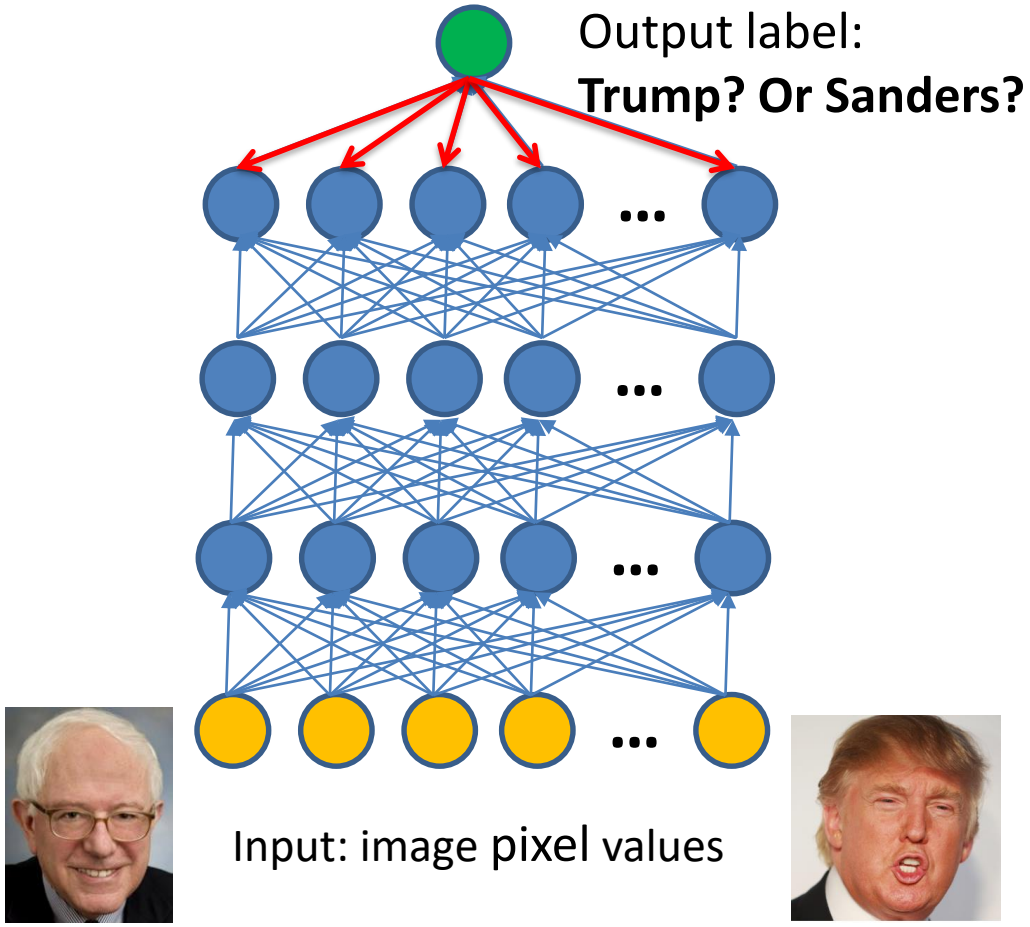
$y = (i_1 + i_2)$ when $(i_1 + i_2) < 1$
$= 1$ when $(i_1 + i_2) >= 1$





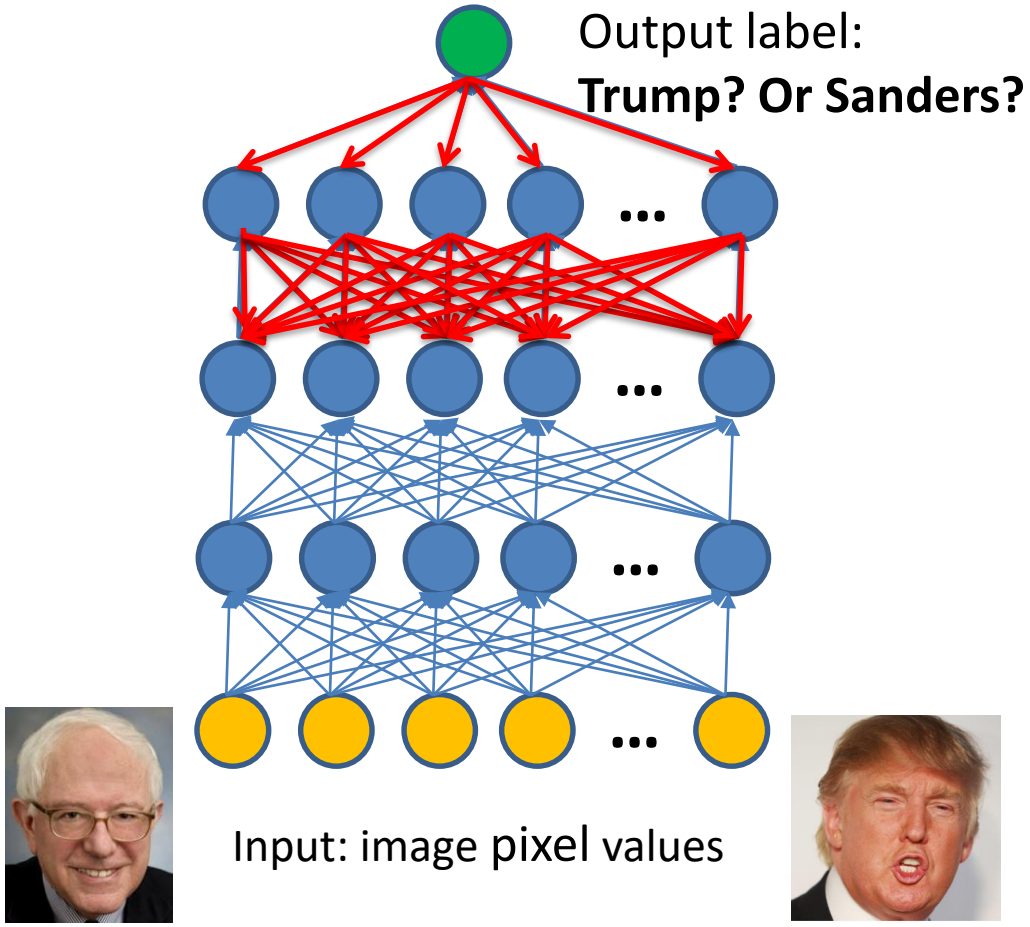
How can we compute importance scores?

Backpropagate contributions



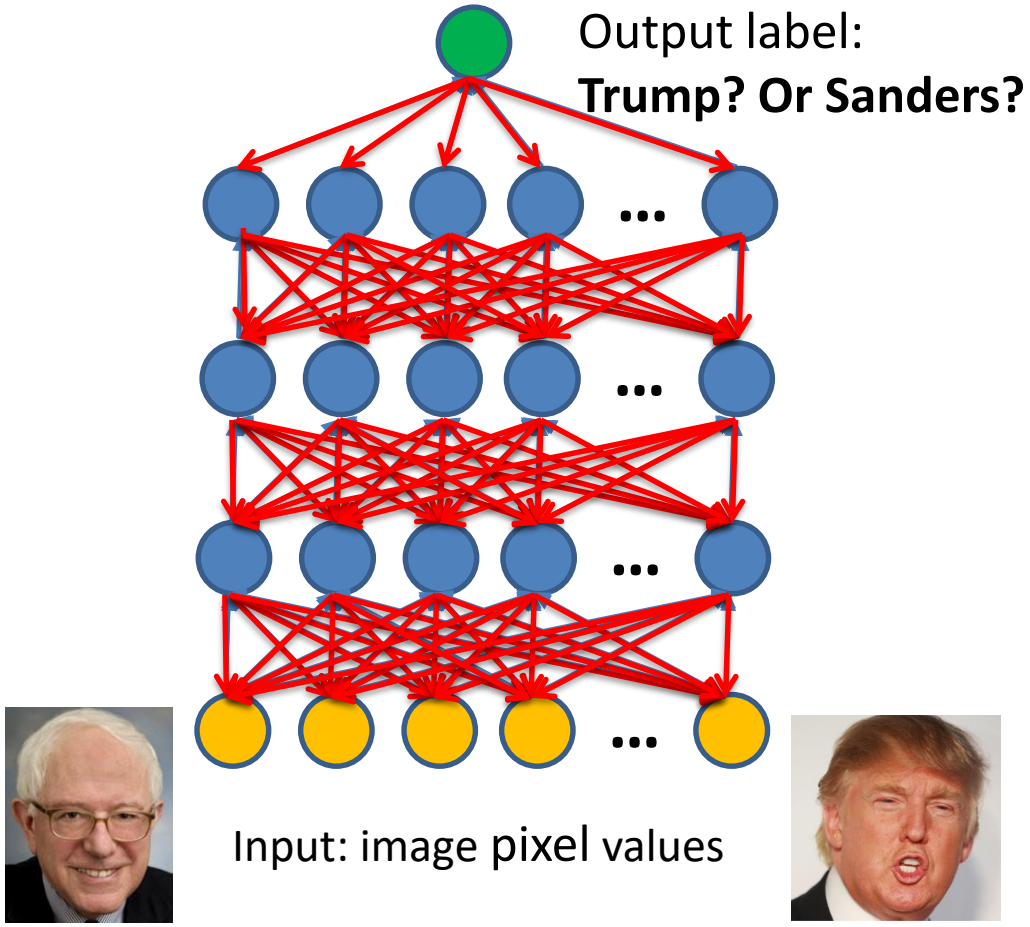
How can we compute importance scores?

Backpropagate contributions



How can we compute importance scores?

Backpropagate contributions



Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization

Forward pass
(bias=0 for all
conv neurons)

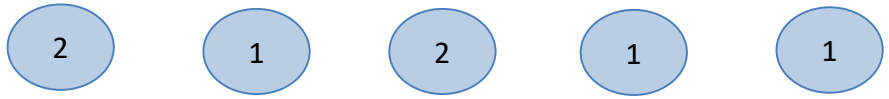
Deconv

Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization

Forward pass
(bias=0 for all
conv neurons)

Deconv

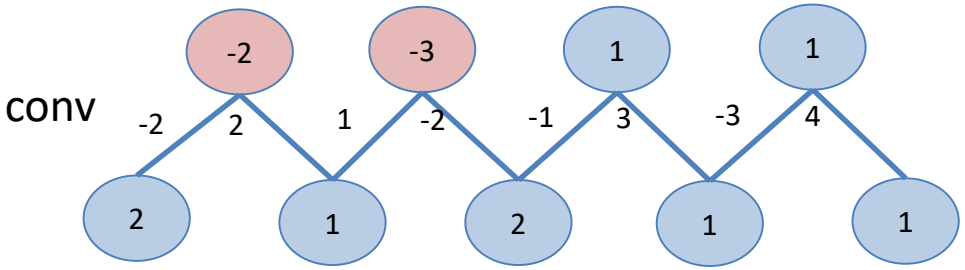


Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization

Forward pass
(bias=0 for all
conv neurons)

Deconv

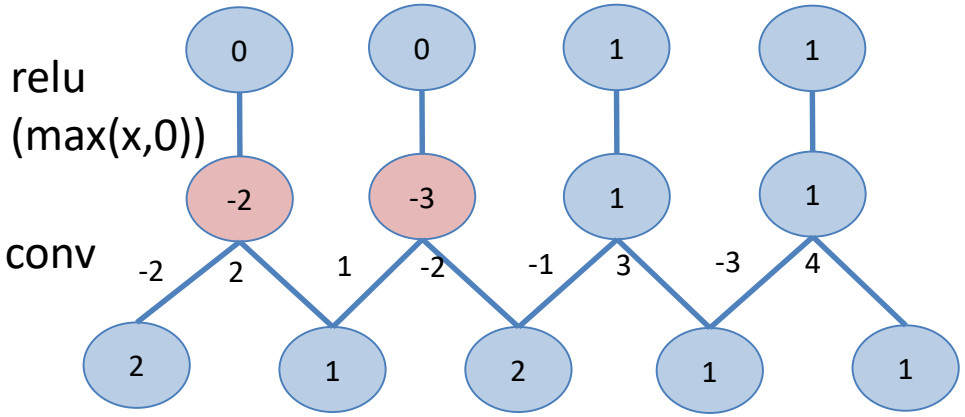


Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization

Forward pass
(bias=0 for all
conv neurons)

Deconv

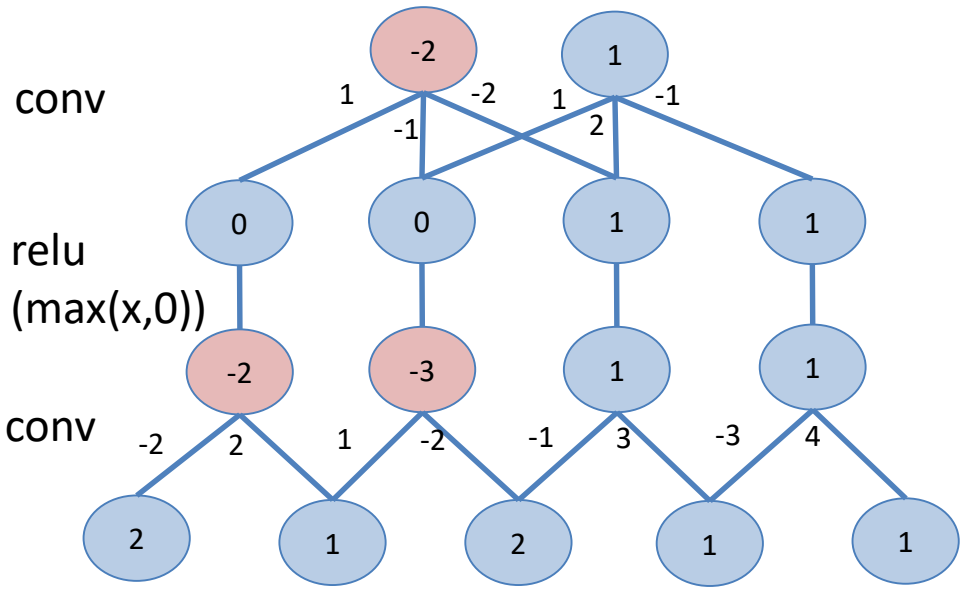


Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization

Forward pass
(bias=0 for all
conv neurons)

Deconv

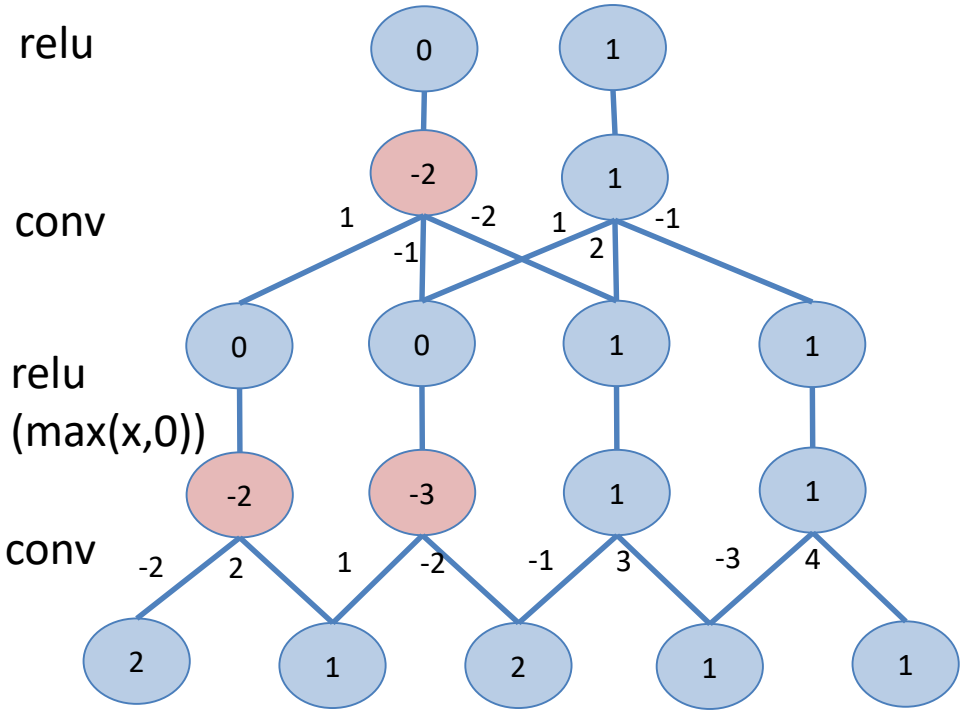


Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization

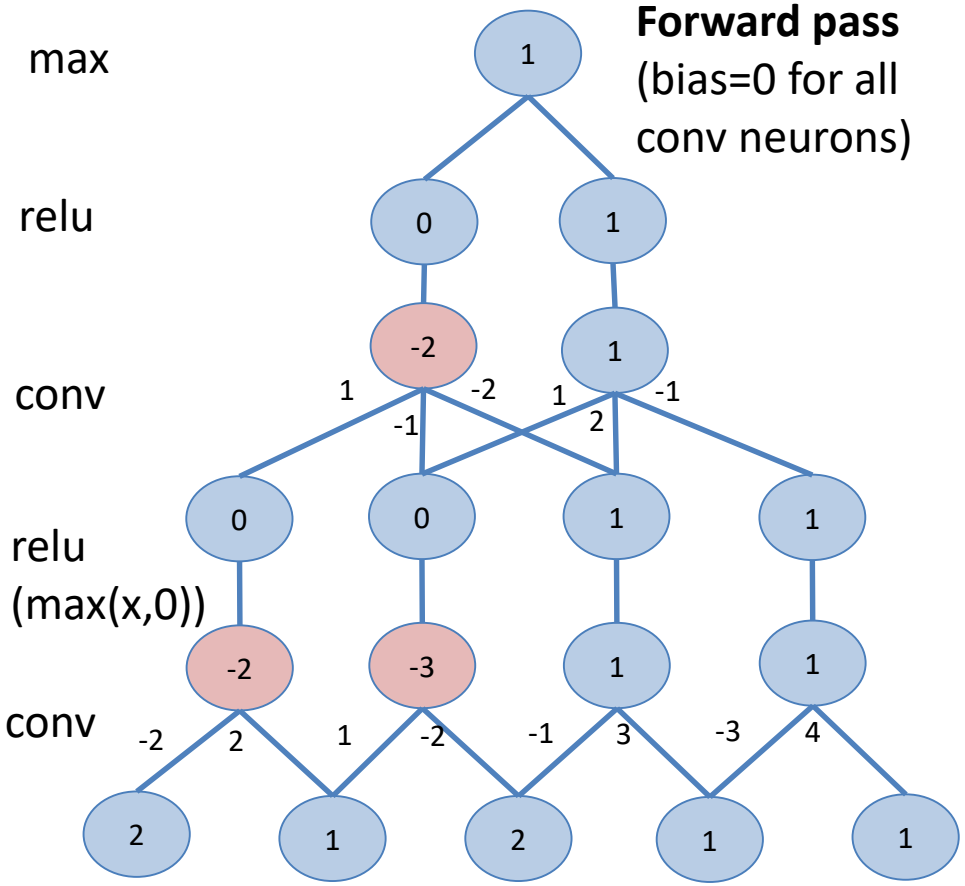
Forward pass
(bias=0 for all
conv neurons)

Deconv



Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization

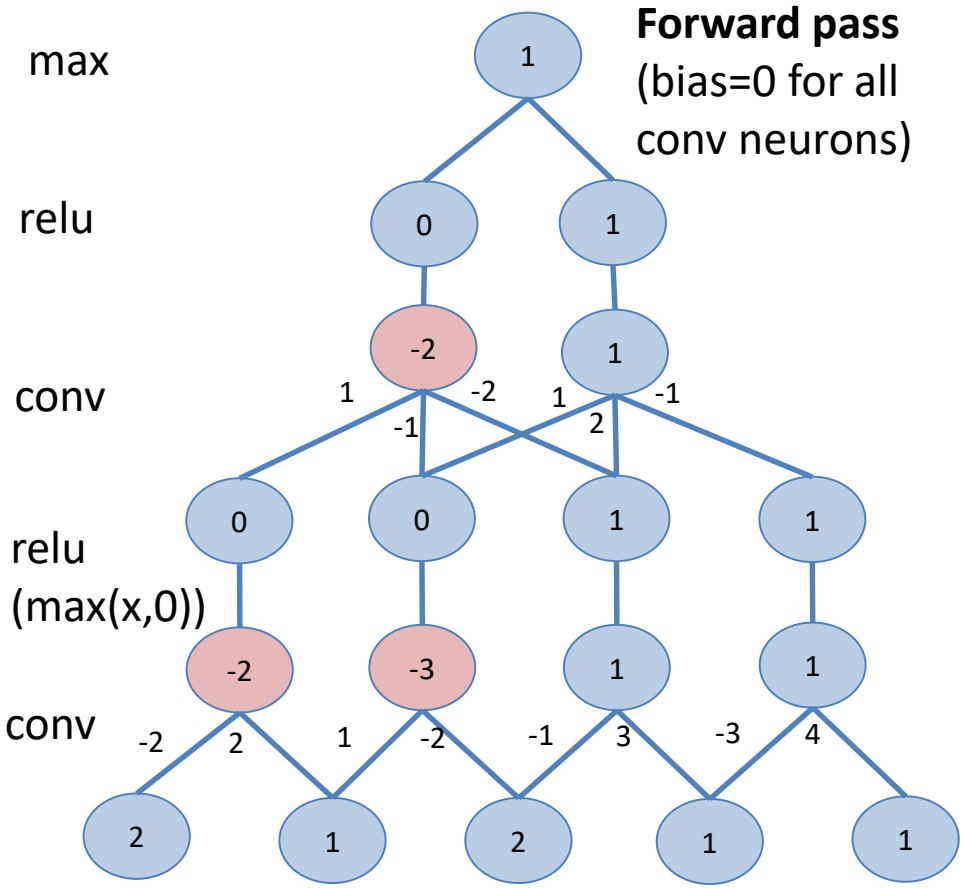


Forward pass
(bias=0 for all
conv neurons)

Deconv

Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization

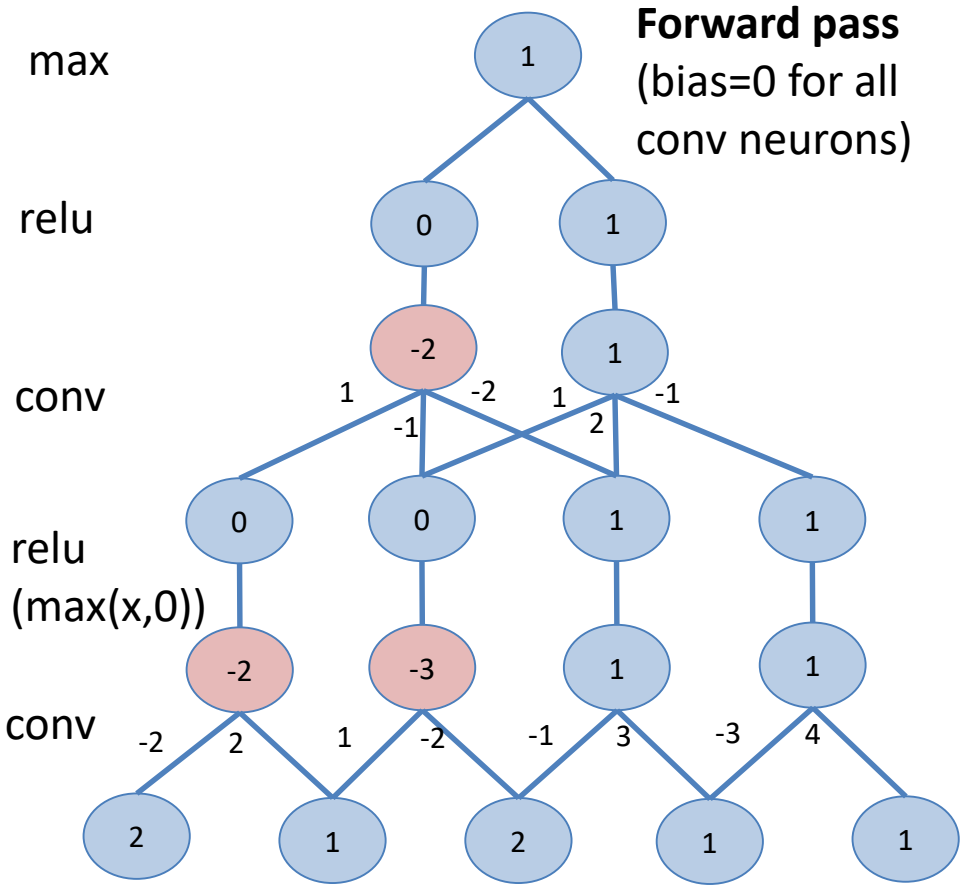


Forward pass
(bias=0 for all
conv neurons)

Deconv

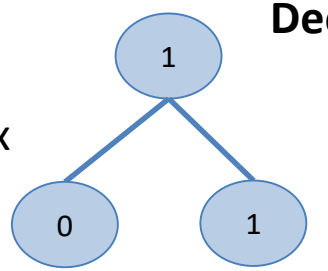
Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization



Forward pass
(bias=0 for all conv neurons)

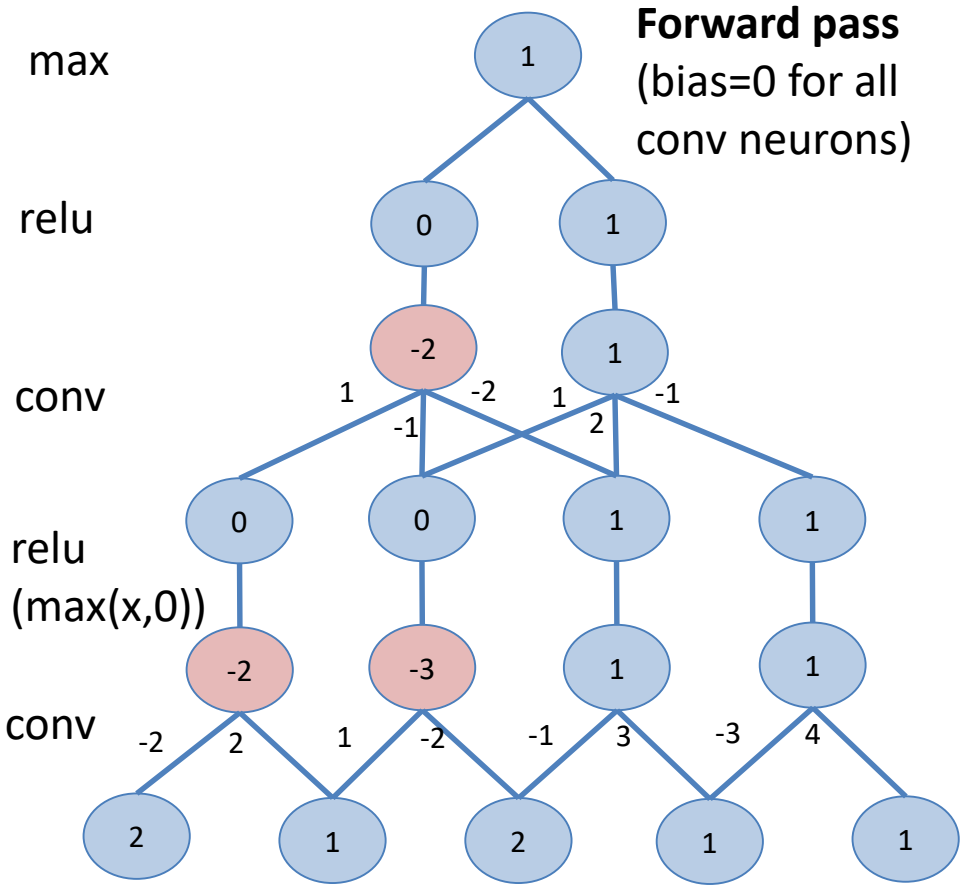
"switch" to correct max

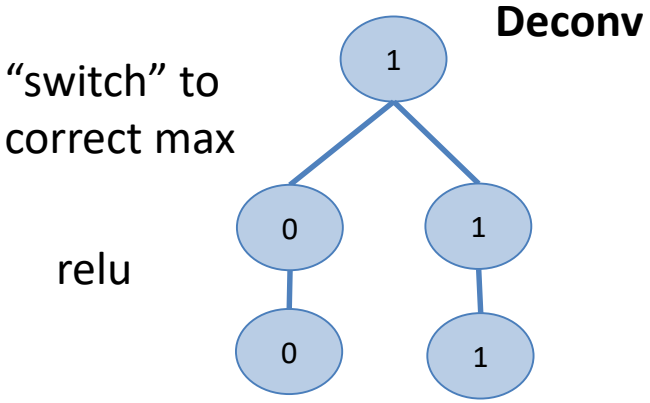


Deconv

Idea 1: deconvolutional nets

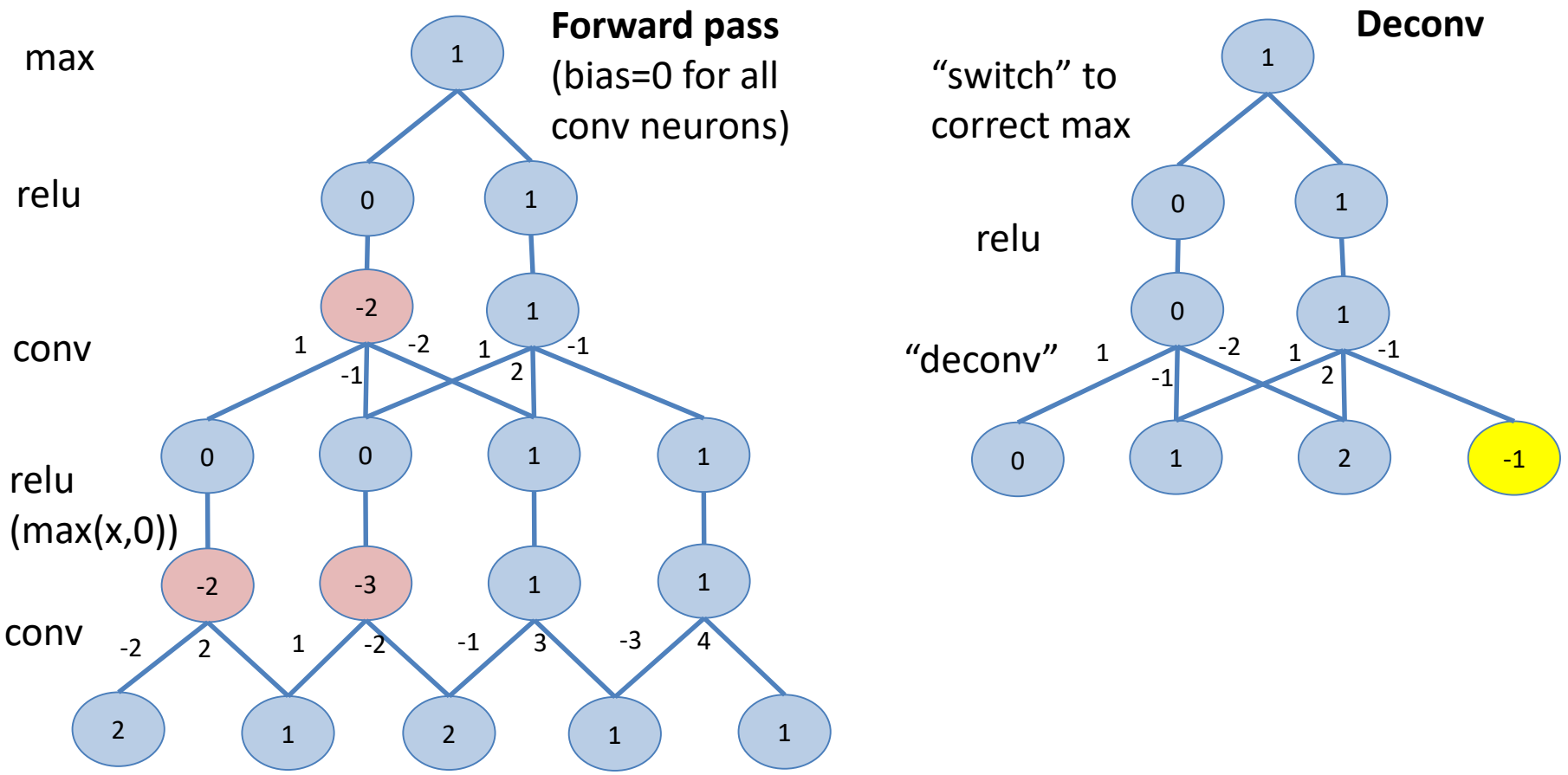
- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization





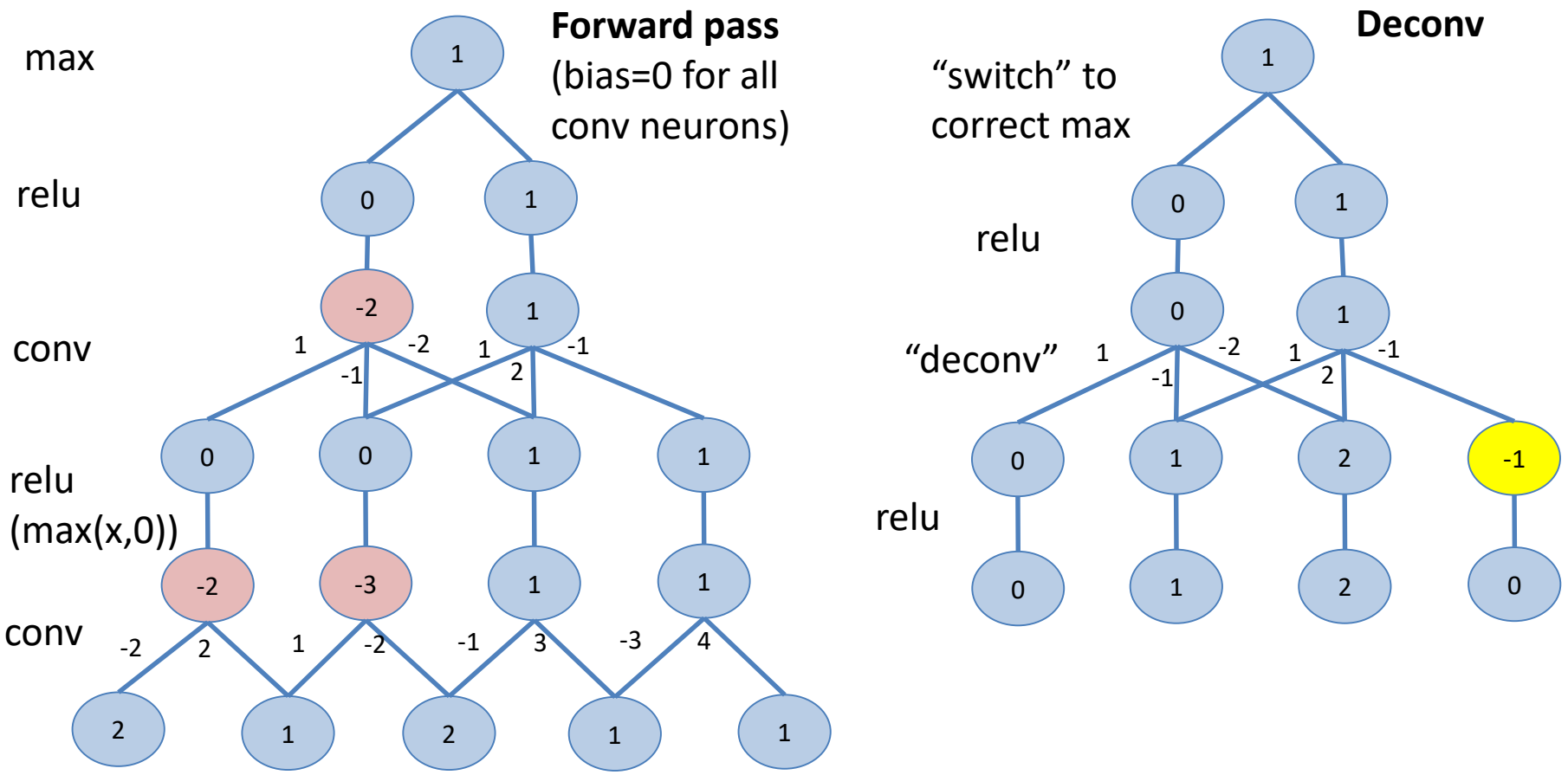
Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization



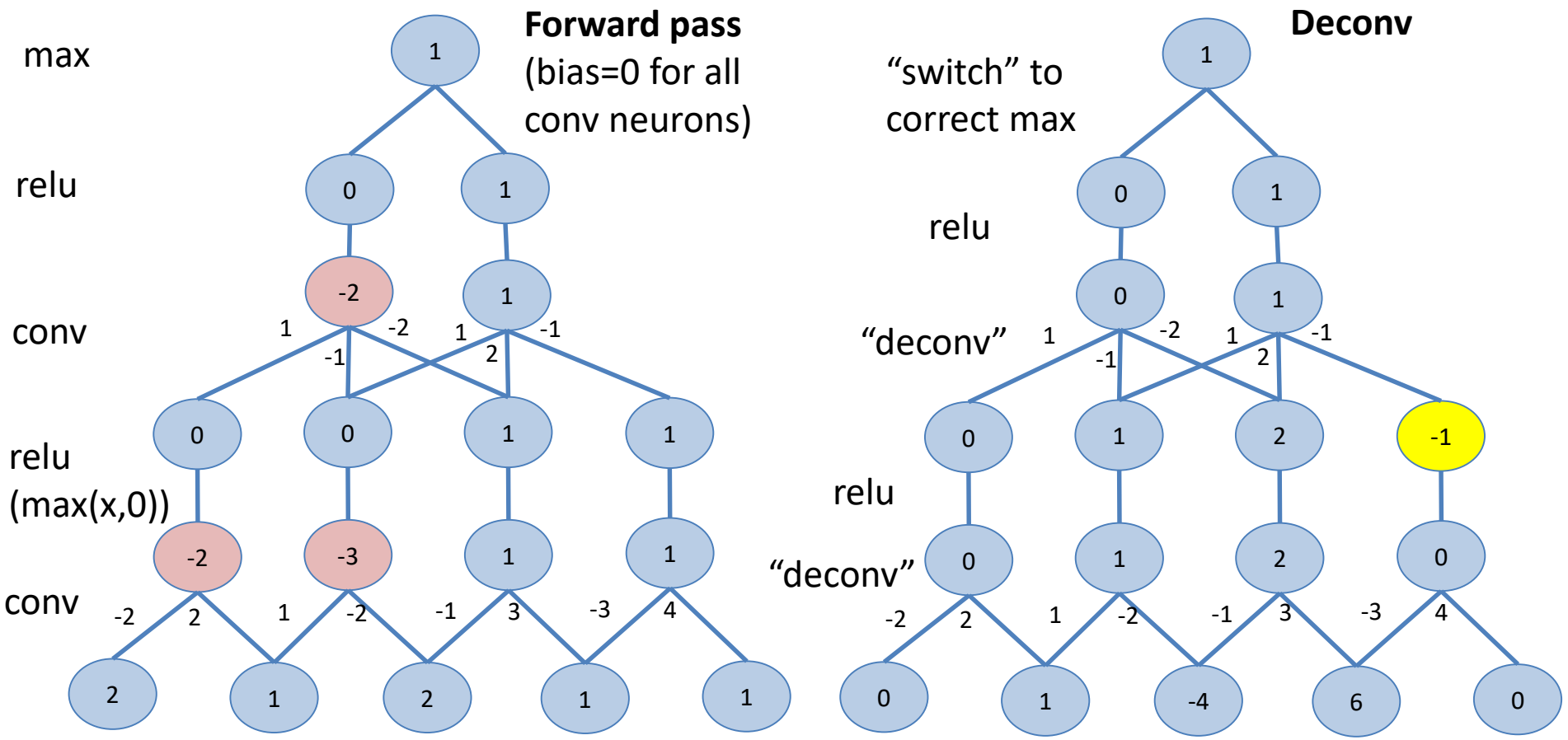
Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization

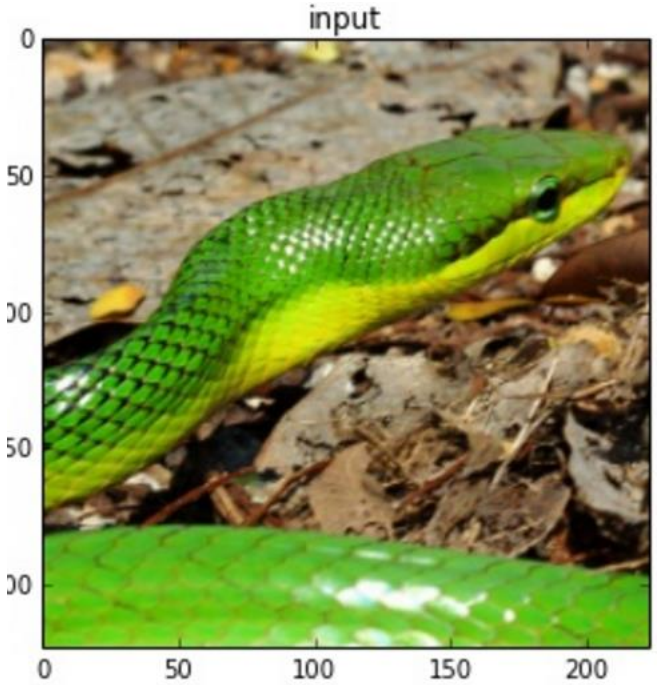


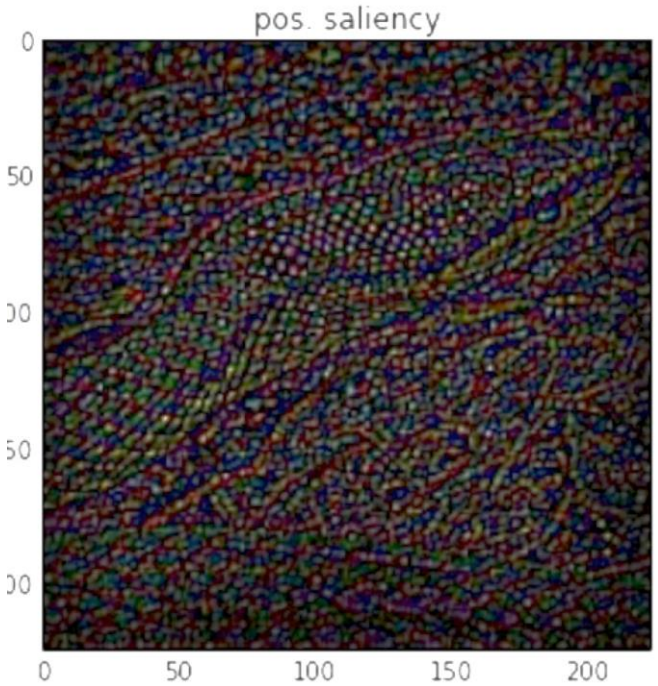
Idea 1: deconvolutional nets

- Originally applied as unsupervised learning:
  - http://www.matthewzeiler.com/pubs/cvpr2010/cvpr2010.pdf
- Adapted for visualization



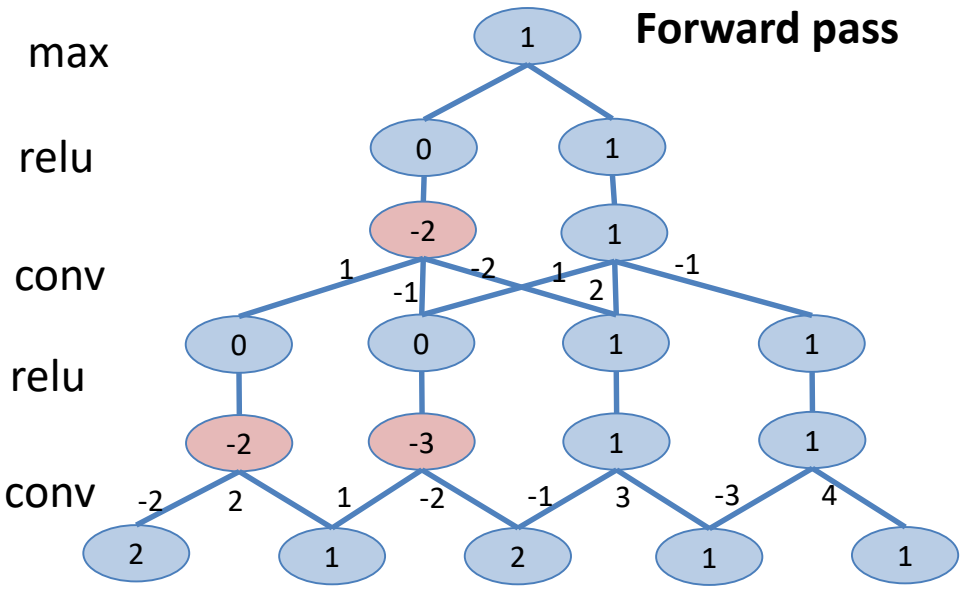
Deconv net: example

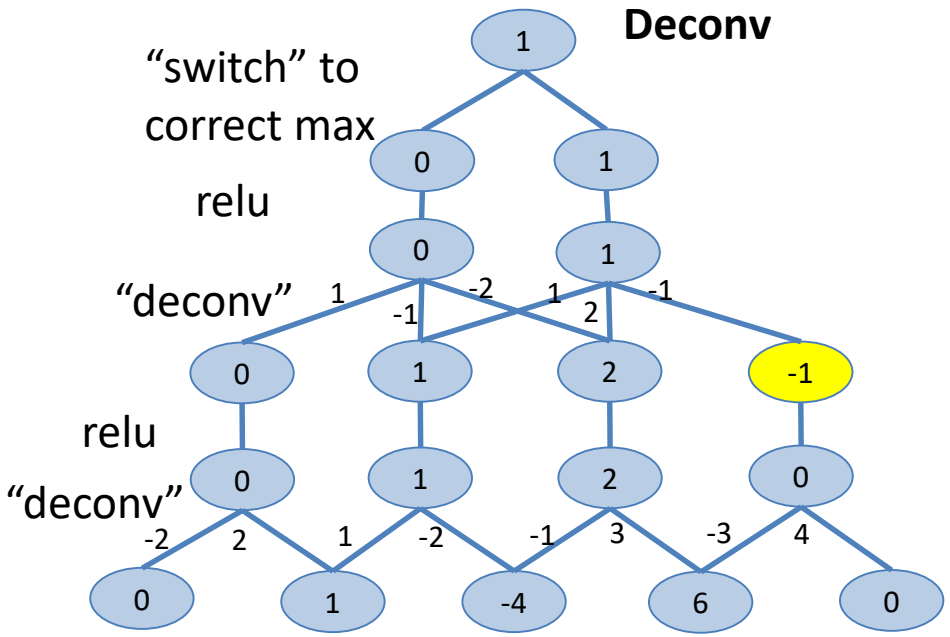




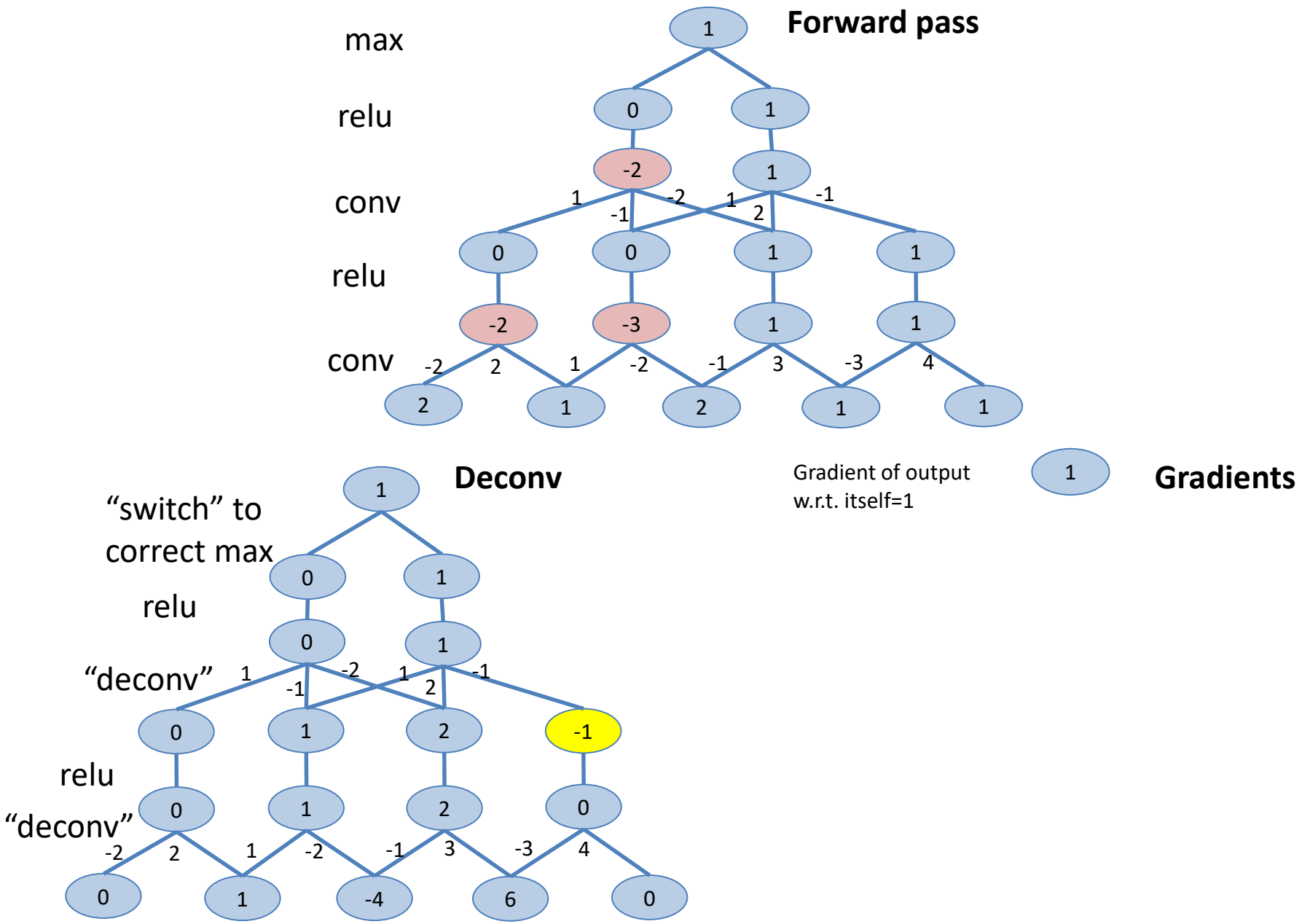
Idea 2: Gradients

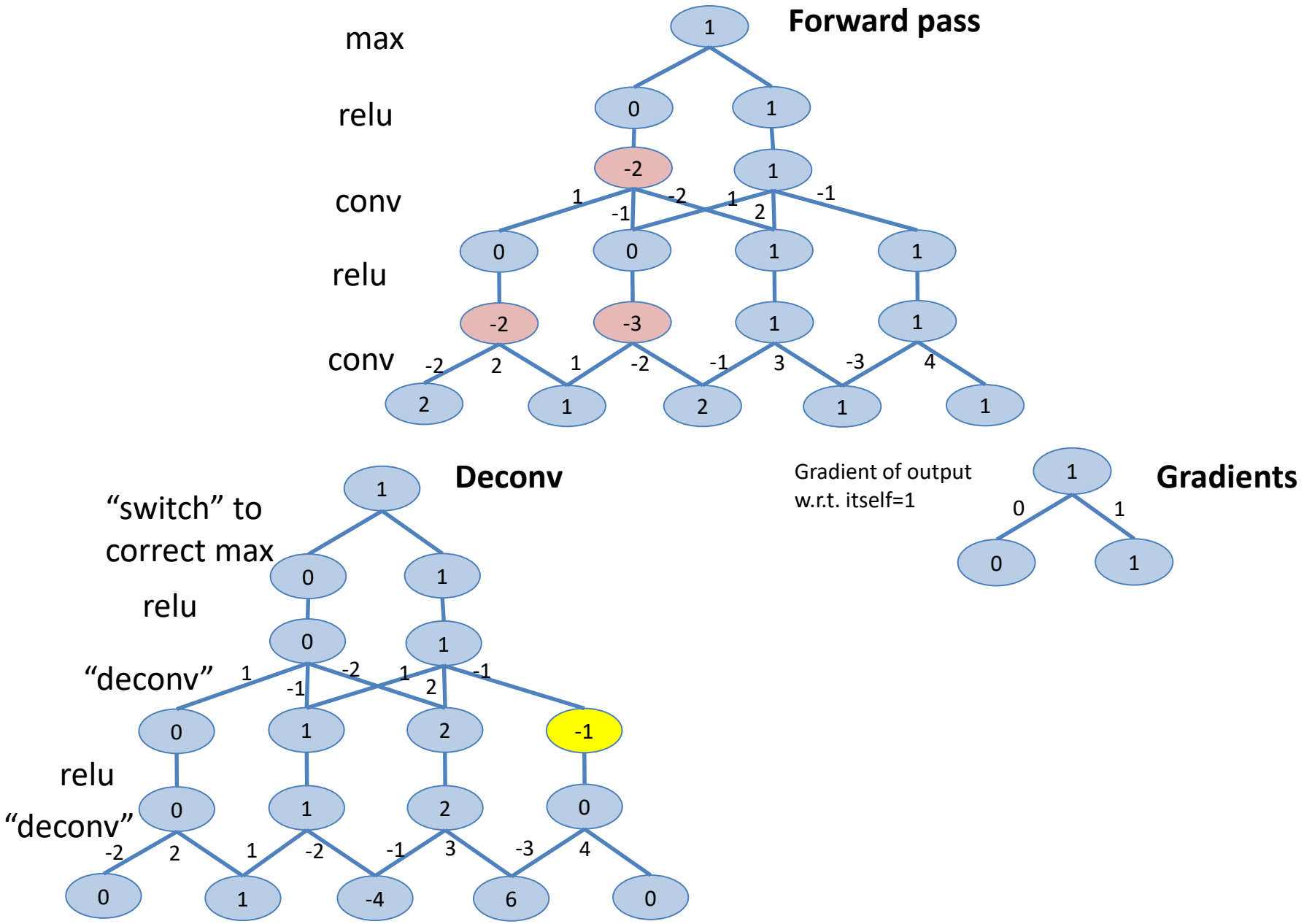
- First proposed by Simonyan et al. Used to improve image segmentation.
- Equivalent to deconv nets, except in handling of ReLUs

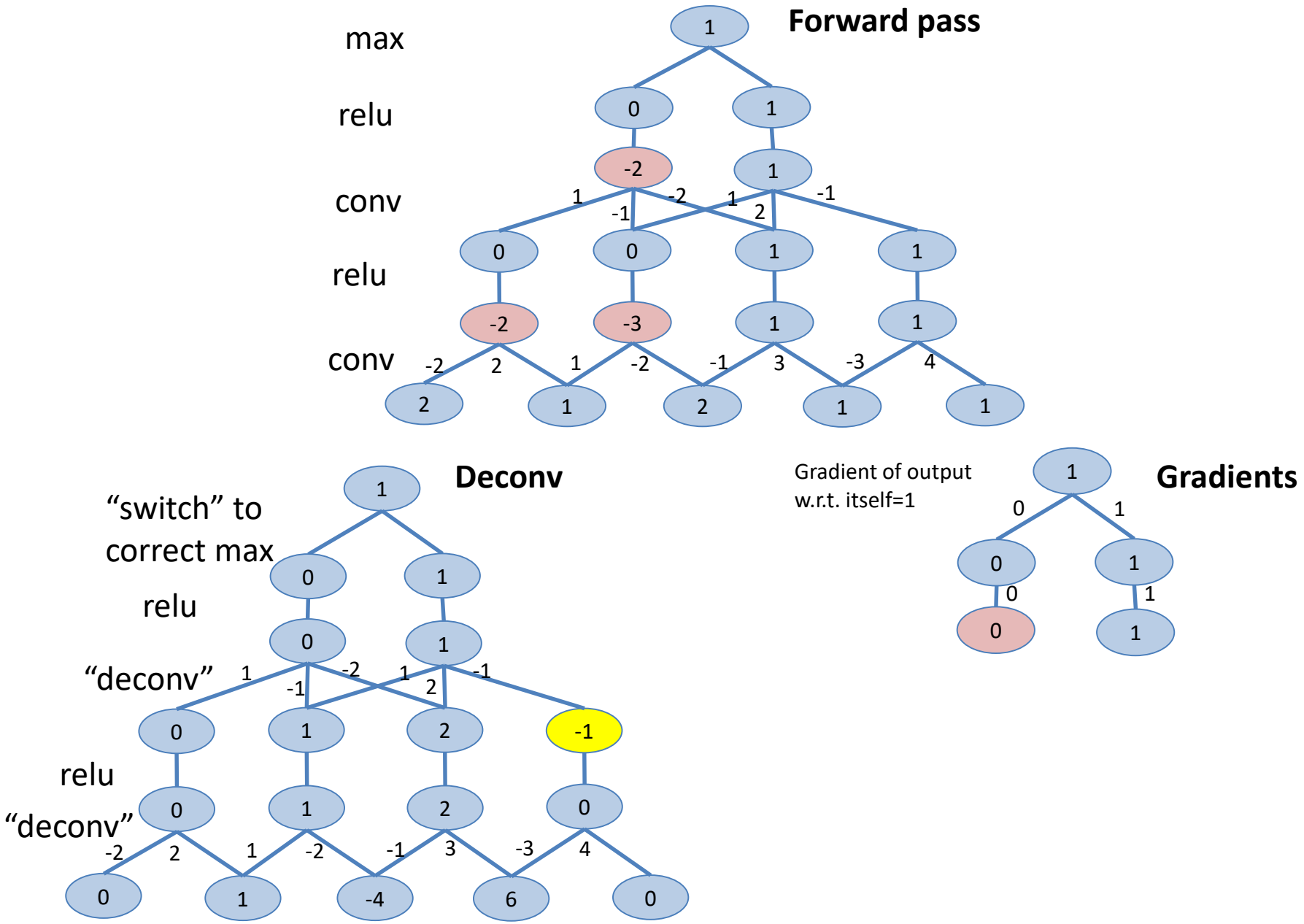

Forward pass
max
relu
conv
relu
conv
1
0
1
-2
1
1
-2
1
-1
-1
2
0
0
1
1
-2
-3
1
1
-2
2
1
-2
-1
3
-3
4
2
1
2
1
1

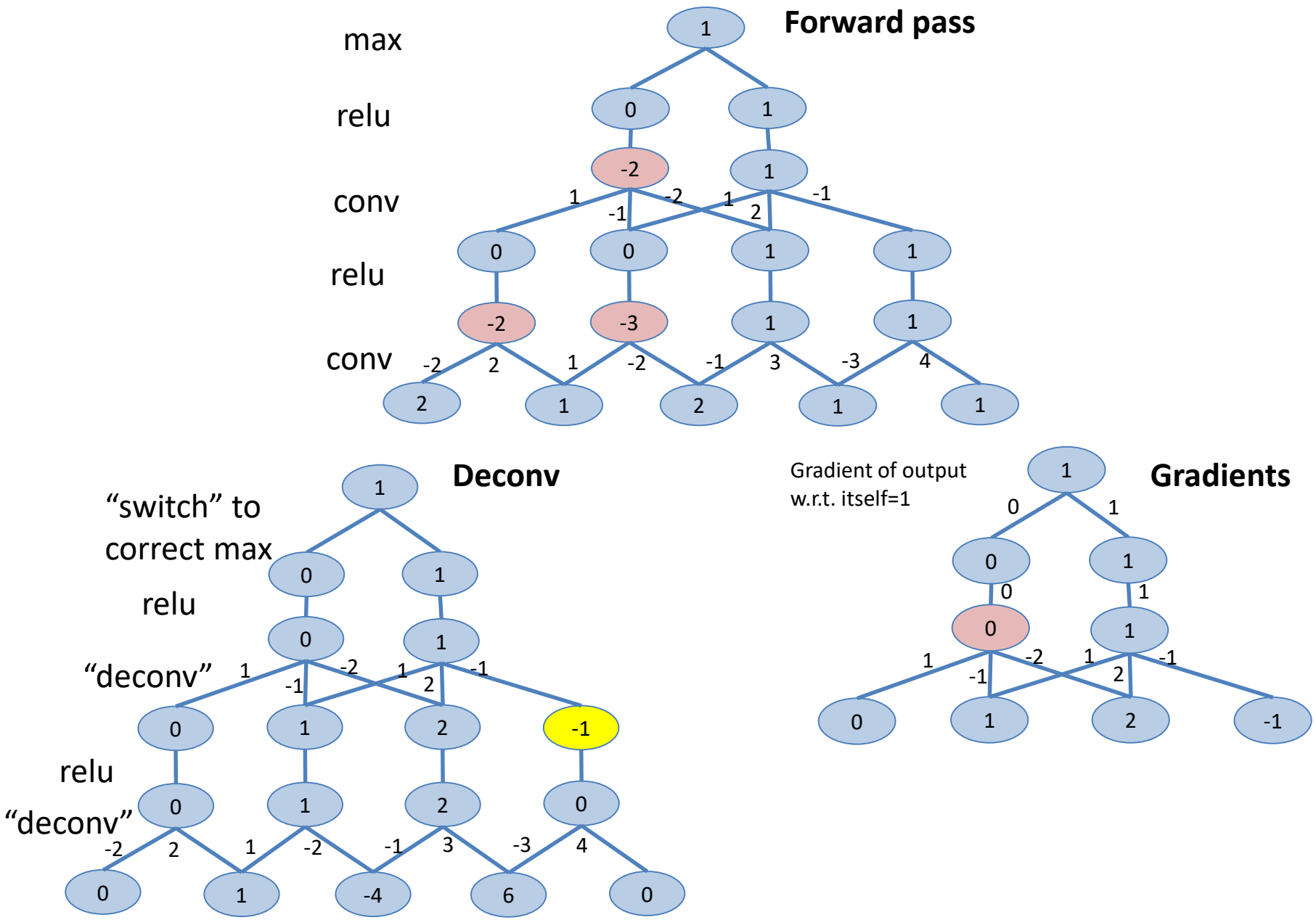

Deconv
"switch" to
correct max
relu
"deconv"
relu
"deconv"
1
0
1
0
1
1
-2
1
-1
-1
2
0
1
2
-1
0
1
2
0
-2
2
1
-2
-1
3
-3
4
0
1
-4
6
0

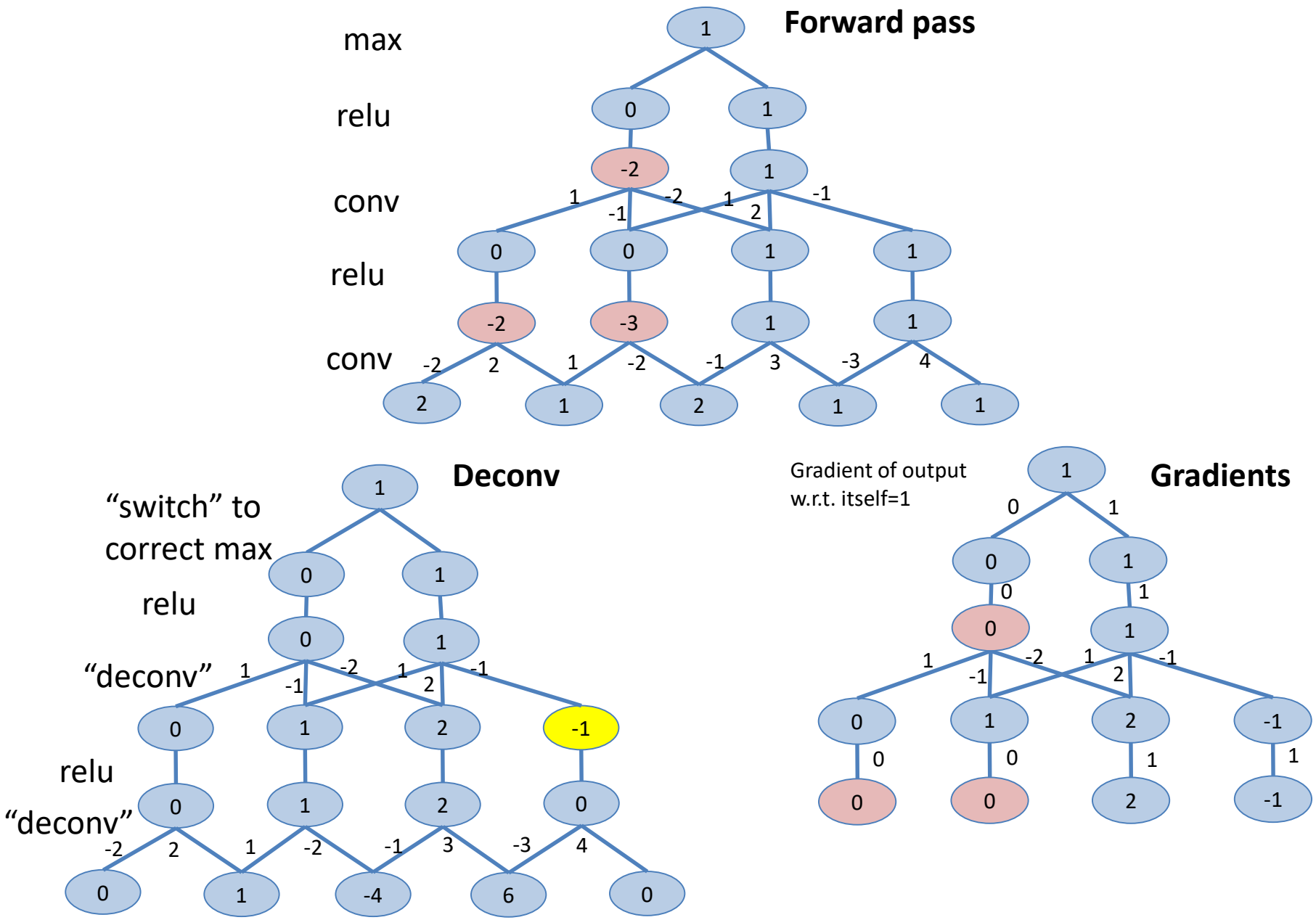
Gradients

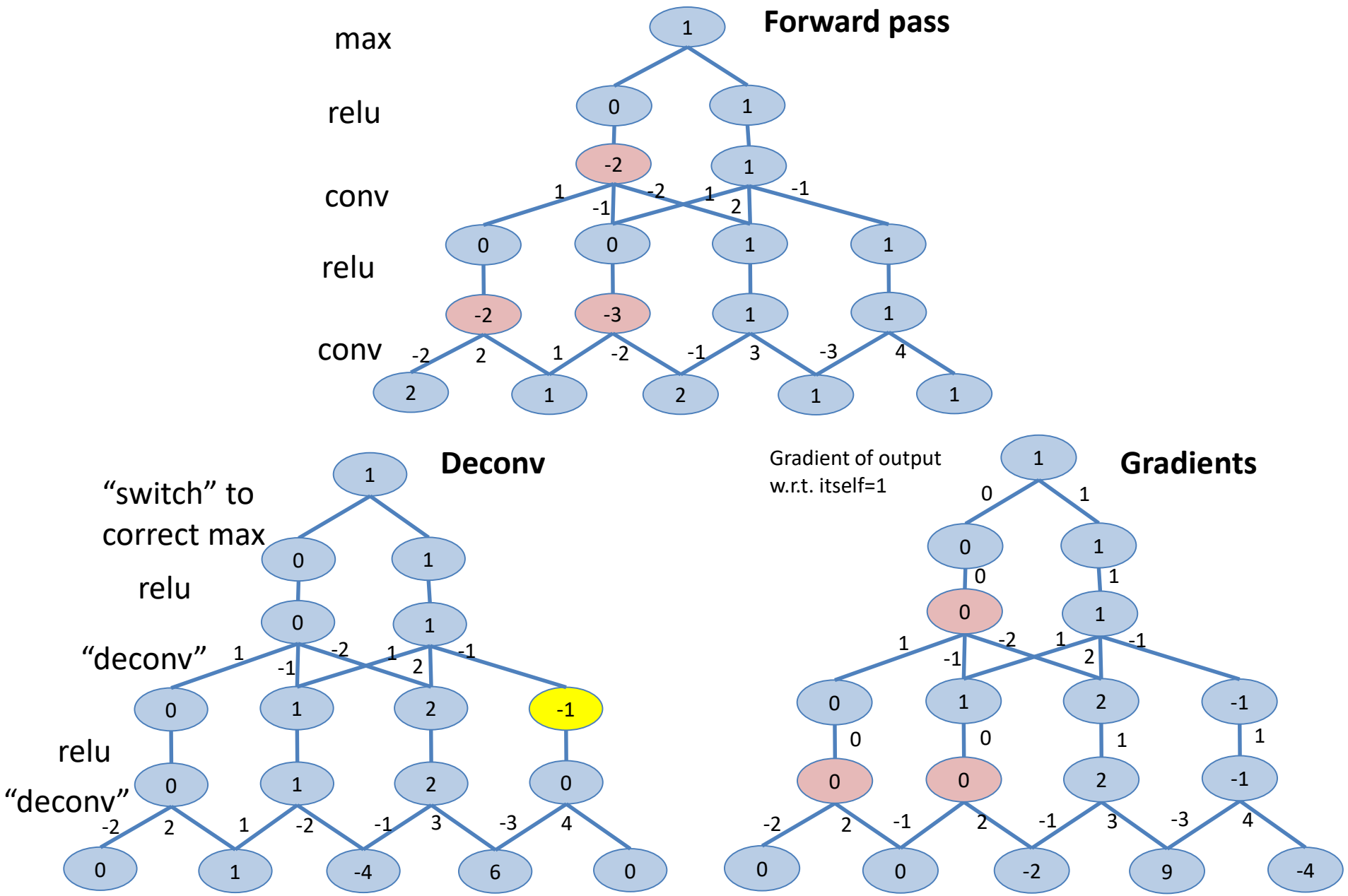

Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to correct max
relu
"deconv"
relu
"deconv"
Gradient of output w.r.t. itself=1
Gradients

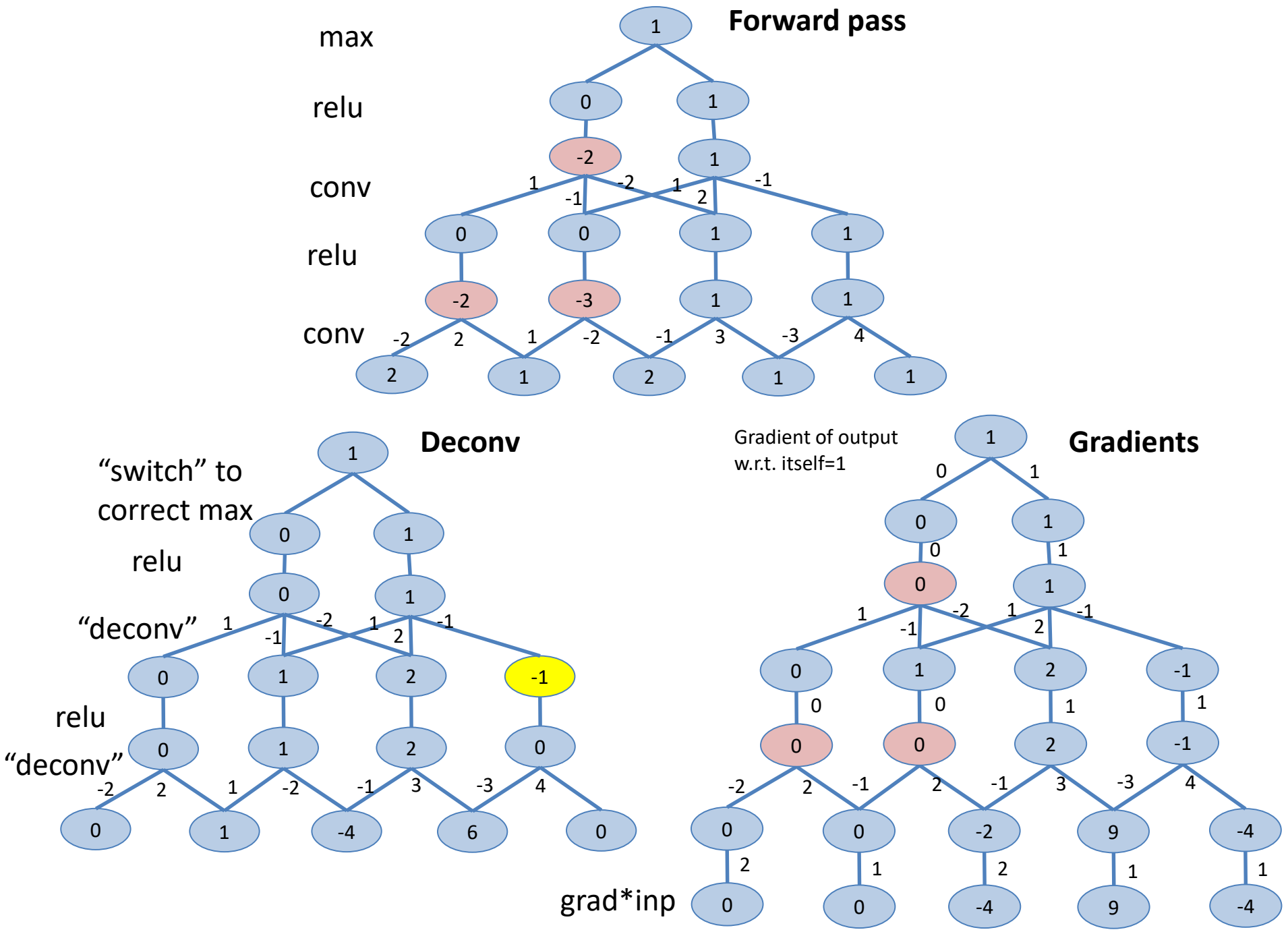

Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to
correct max
relu
"deconv"
relu
"deconv"
Gradient of output
w.r.t. itself=1
Gradients

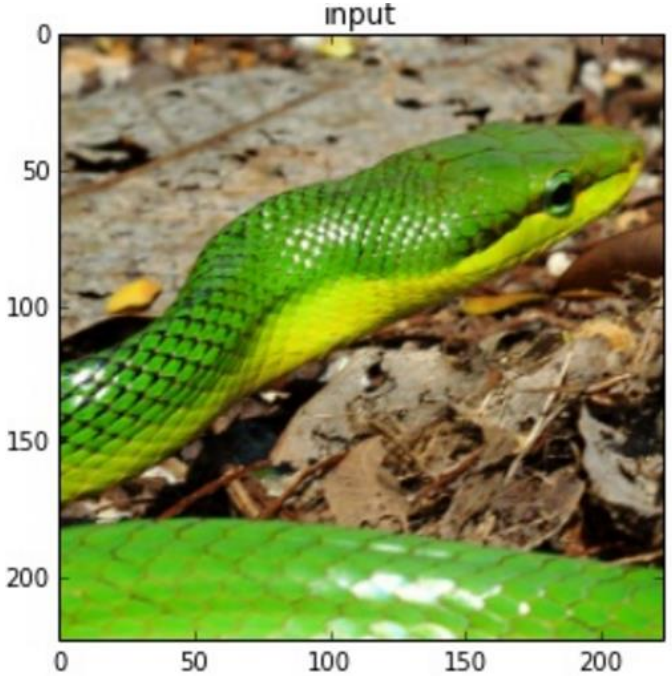

Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to
correct max
relu
"deconv"
relu
"deconv"
Gradient of output
w.r.t. itself=1
Gradients


Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to
correct max
relu
"deconv"
relu
"deconv"
Gradient of output
w.r.t. itself=1
Gradients

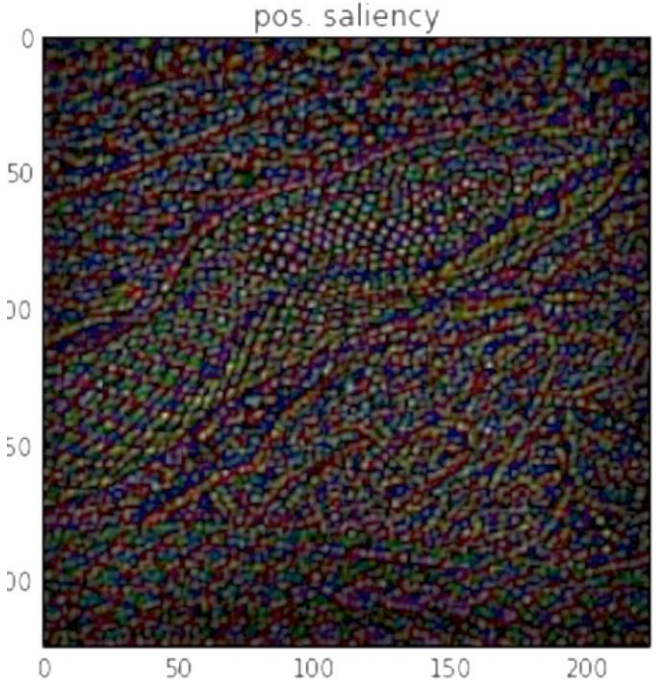

Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to
correct max
relu
"deconv"
relu
"deconv"
Gradient of output
w.r.t. itself=1
Gradients


Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to
correct max
relu
"deconv"
relu
"deconv"
Gradient of output
w.r.t. itself=1
Gradients

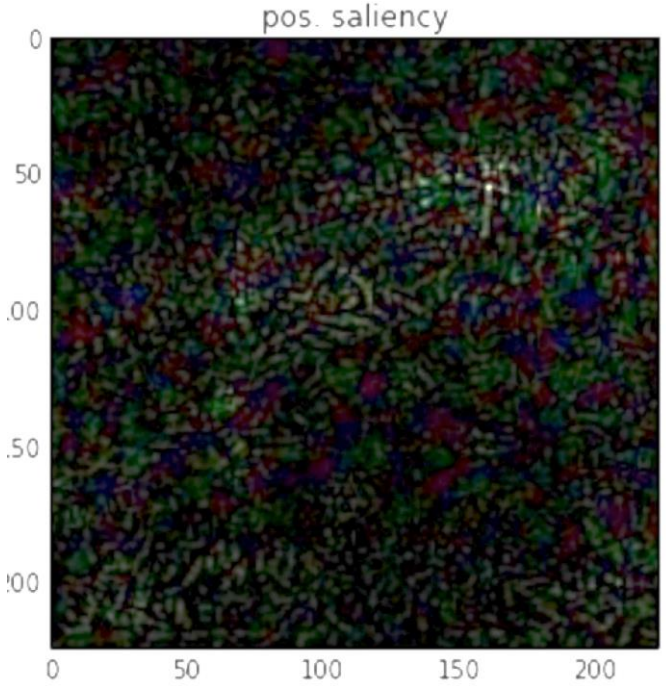

Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to
correct max
relu
"deconv"
relu
"deconv"
Gradient of output
w.r.t. itself=1
Gradients
grad*inp



Deconv

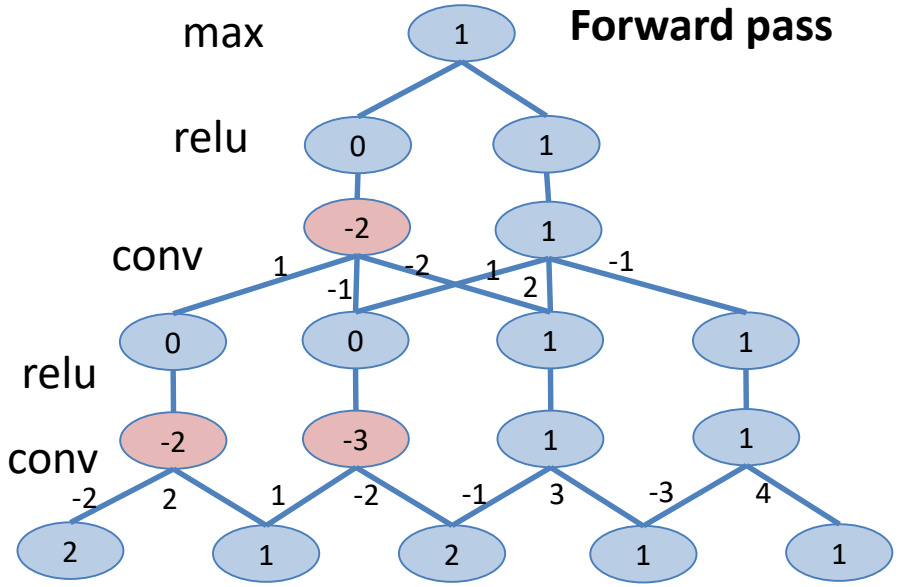


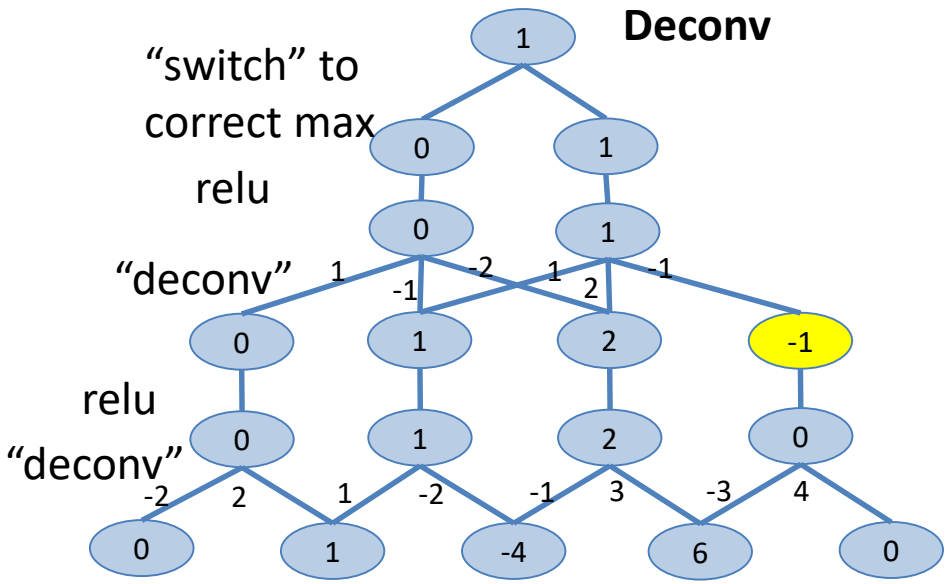
Gradients

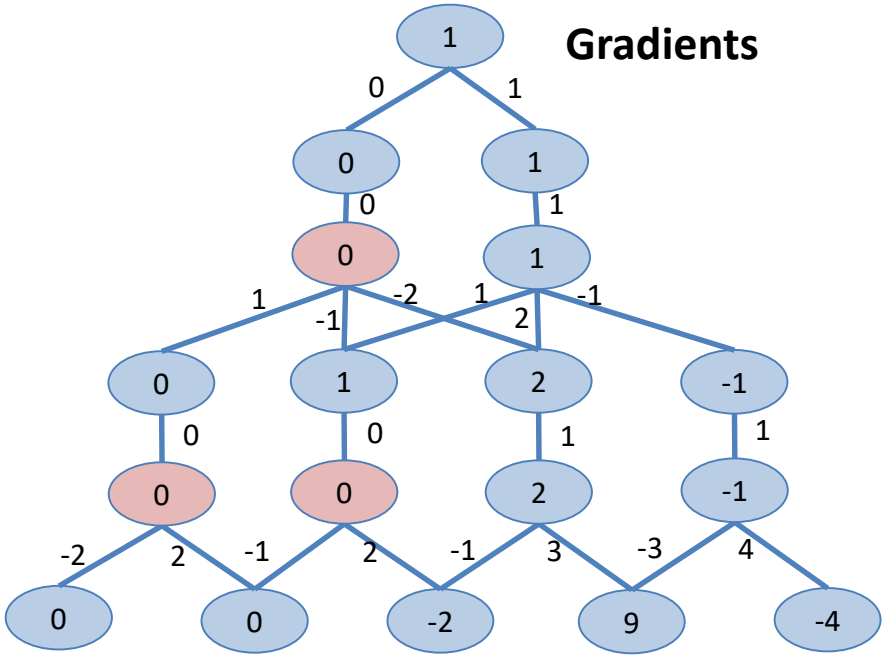


Idea 3: “Guided backpropagation”

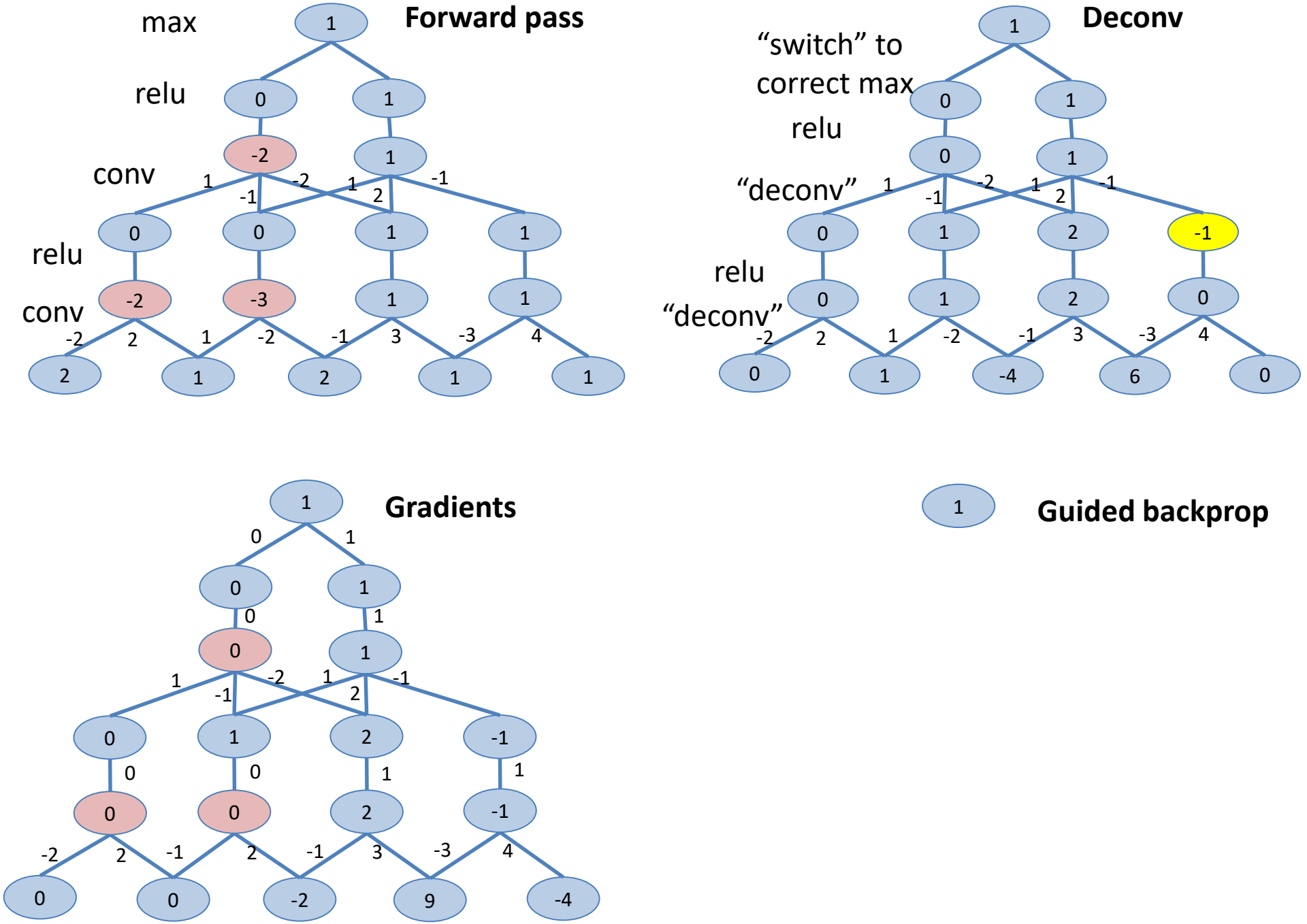
- Deconv seems to have some advantages, gradients seems to have some advantages
- Just combine the two
- When going through ReLUs, zero out gradients when input is < 0 AND when gradient is < 0

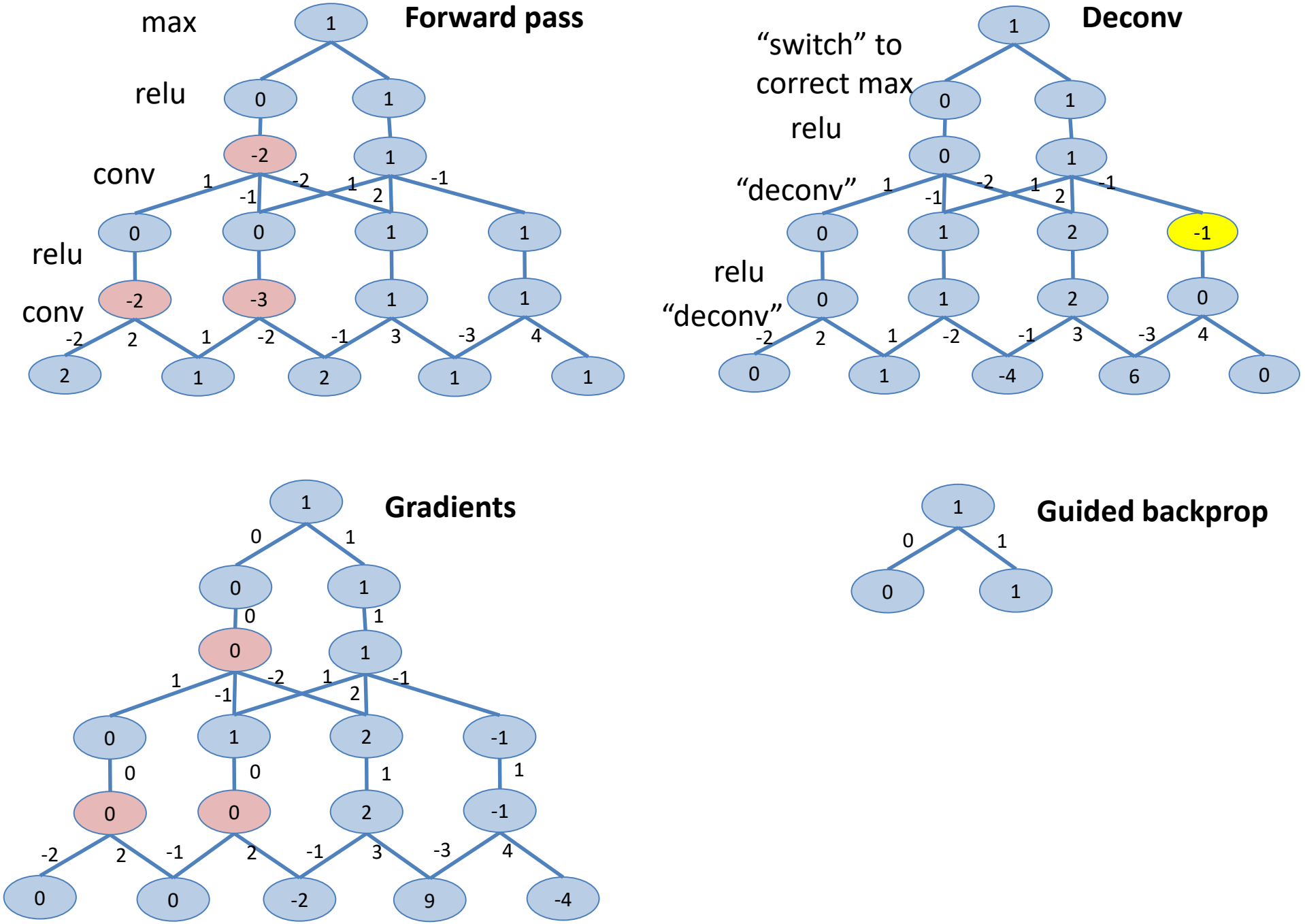


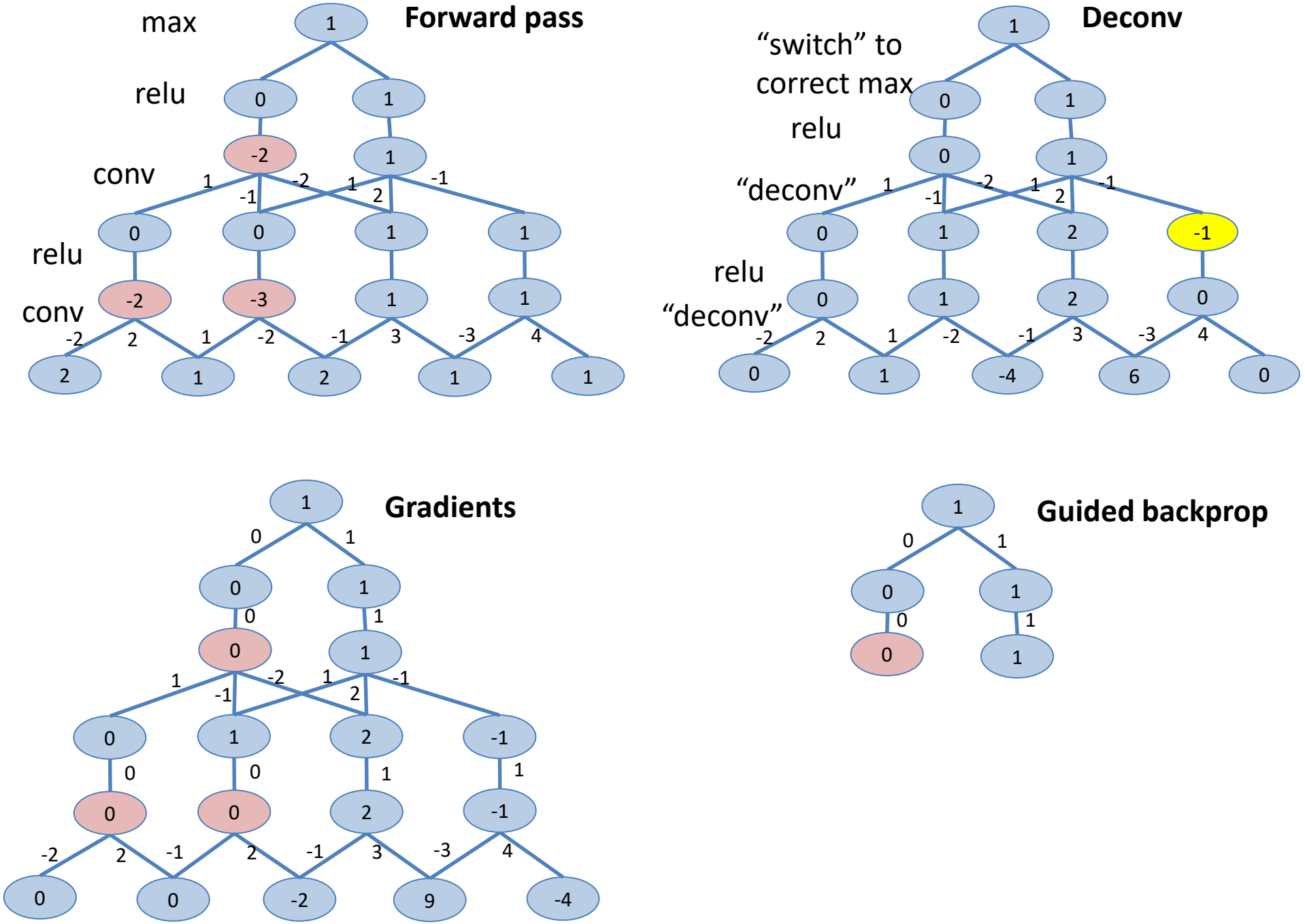


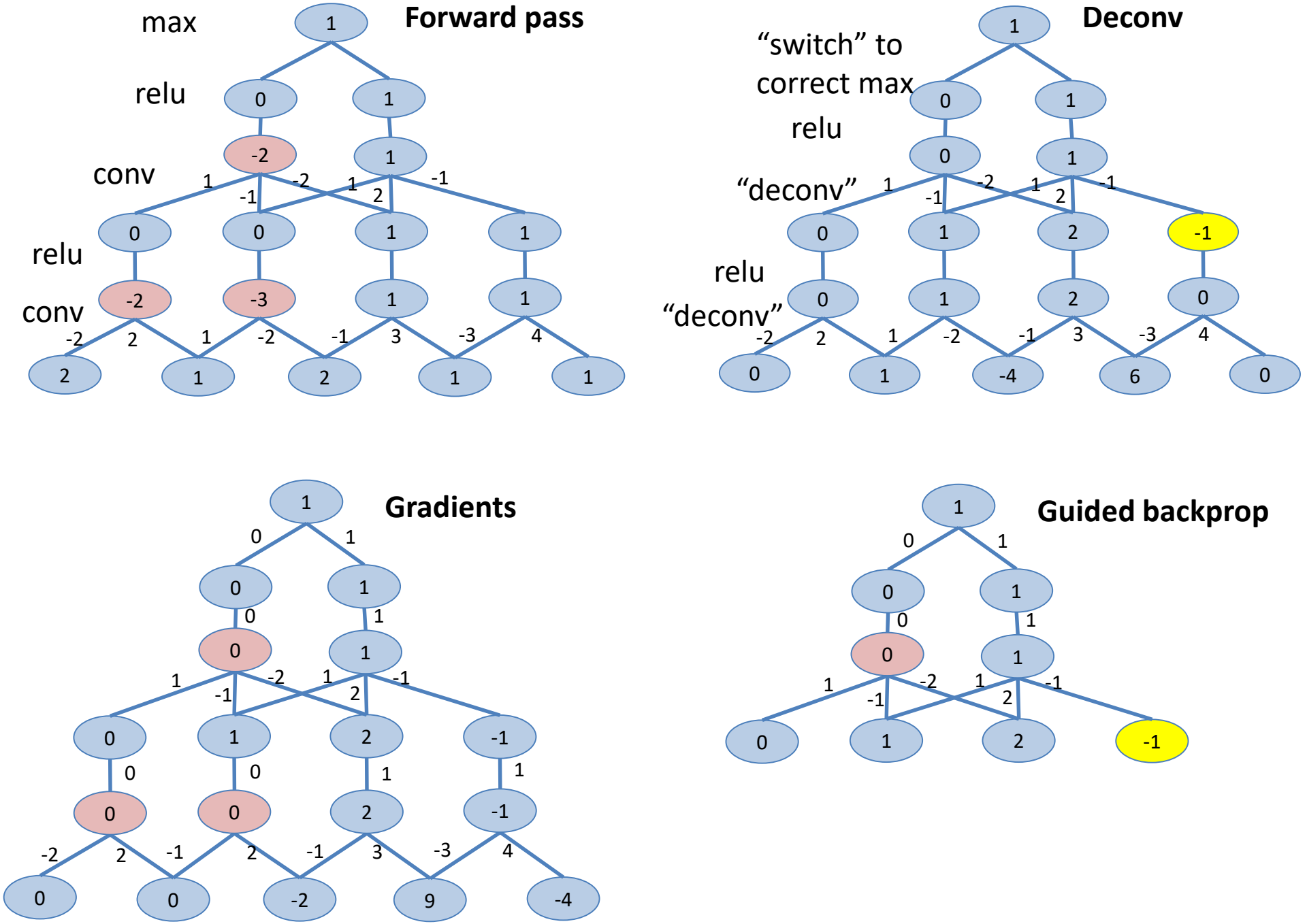


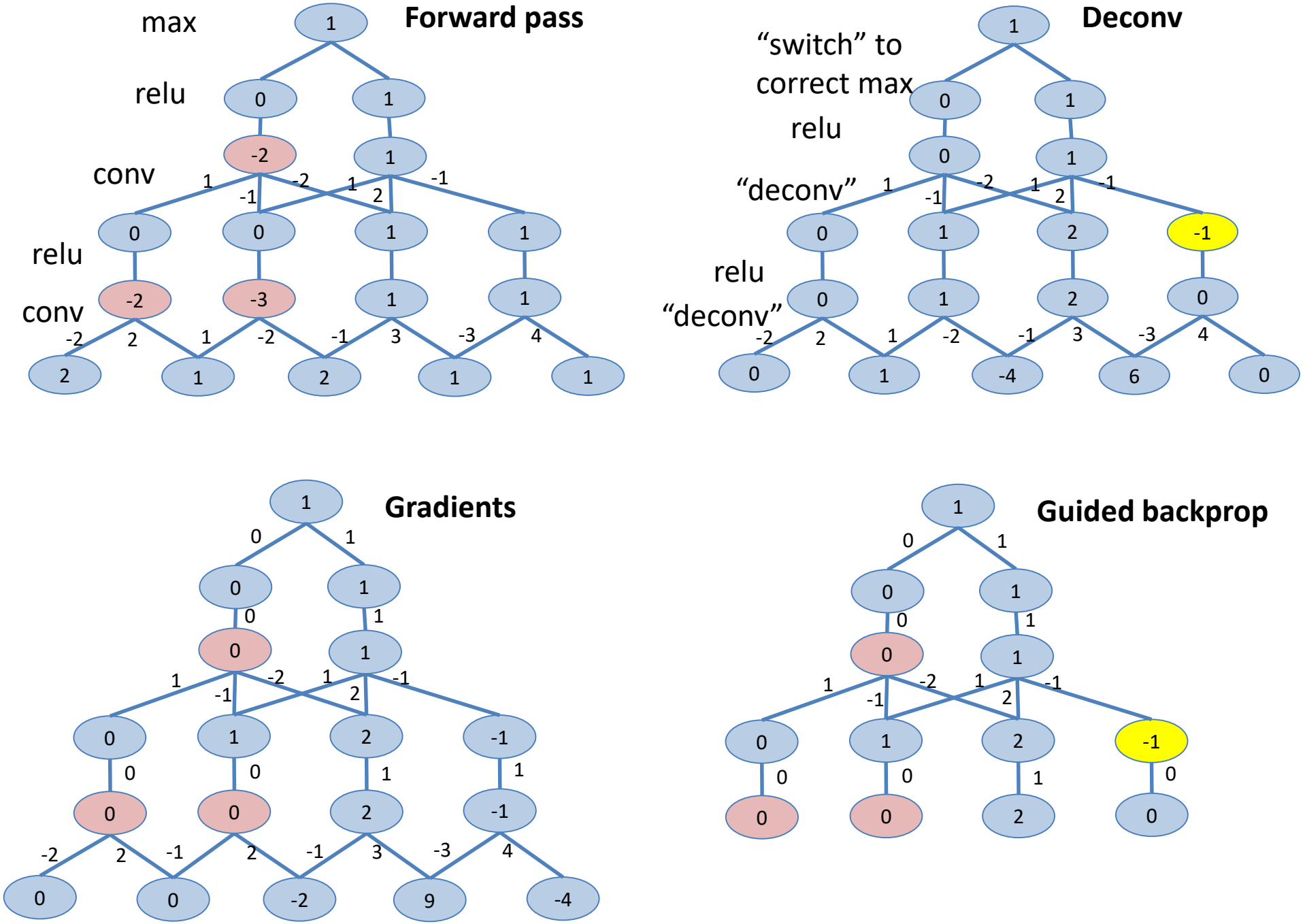
Guided backprop

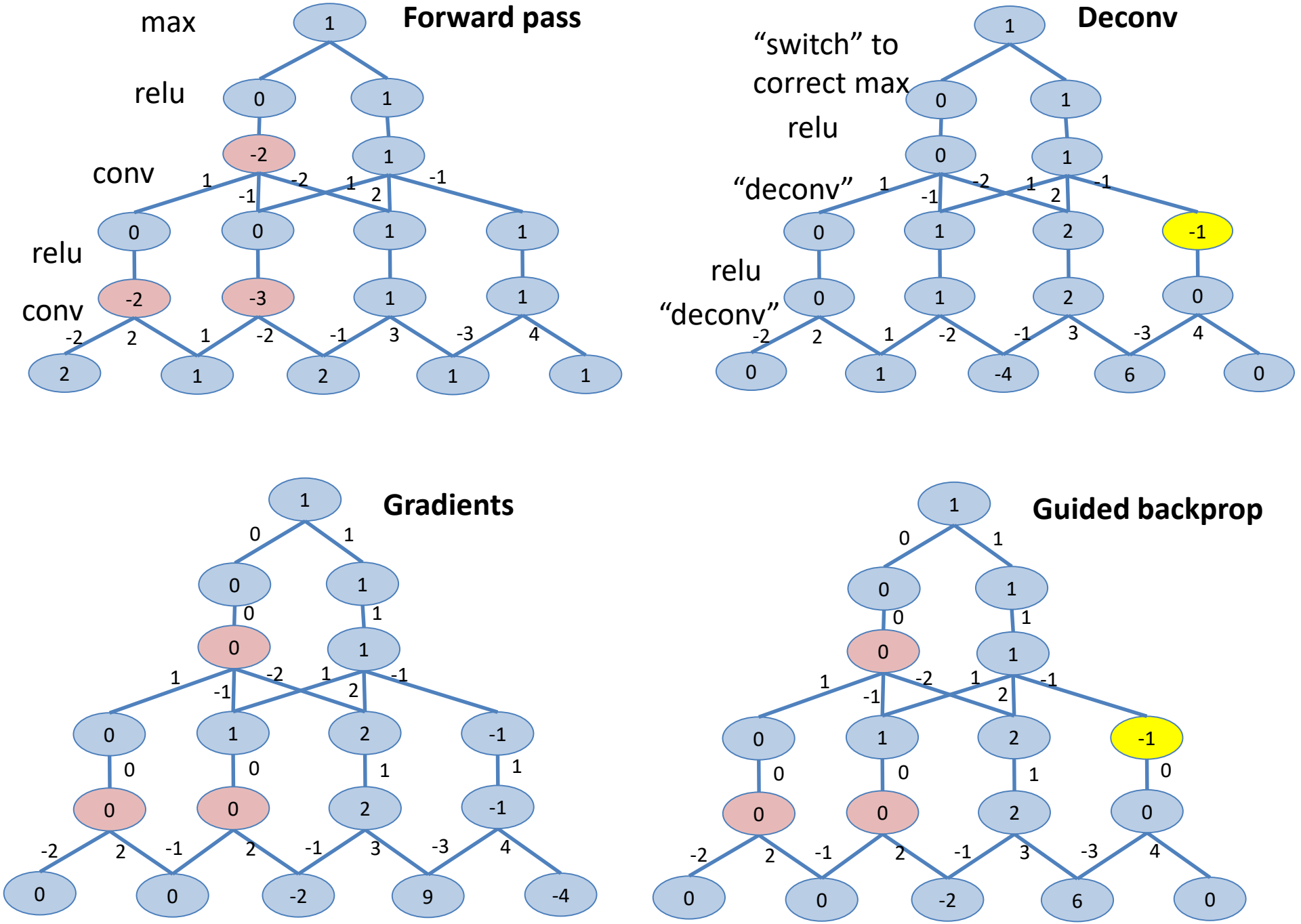

Forward pass
max
relu
conv
relu
conv
Deconv
“switch” to correct max
relu
“deconv”
relu
“deconv”
Gradients
Guided backprop

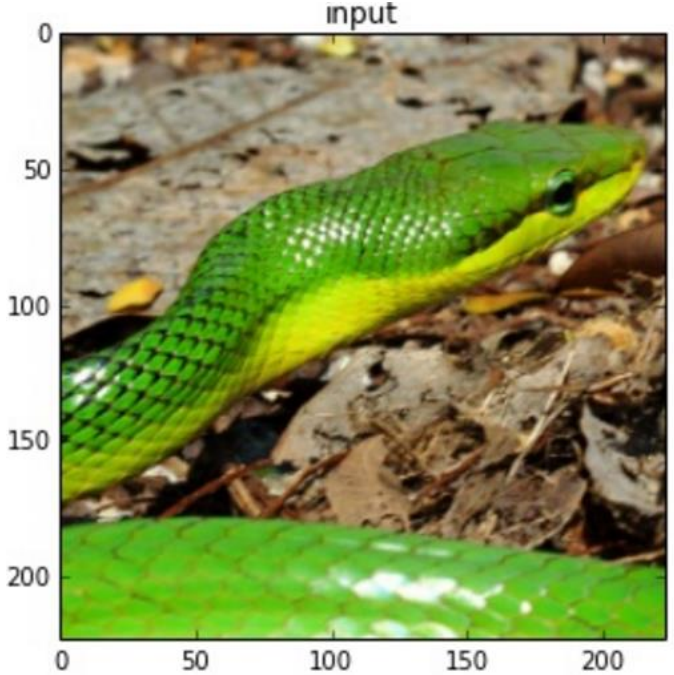

Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to correct max
relu
"deconv"
relu
"deconv"
Gradients
Guided backprop


Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to correct max
relu
"deconv"
relu
"deconv"
Gradients
Guided backprop

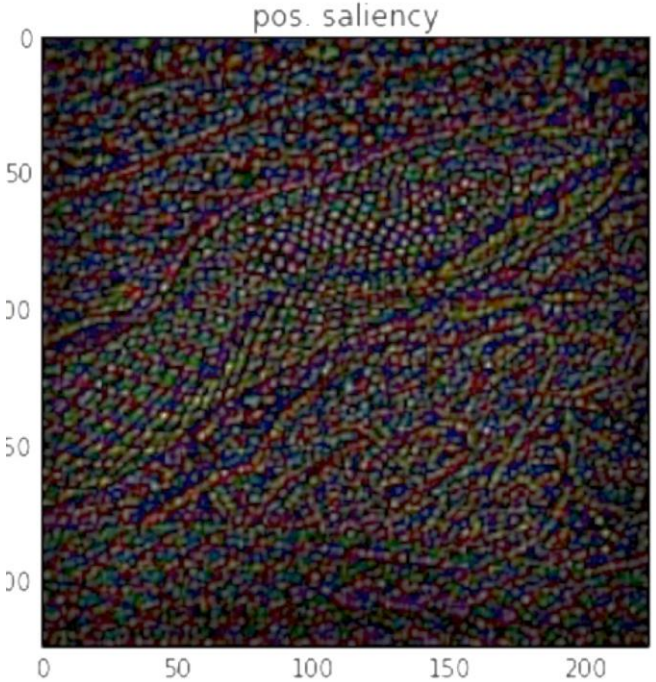

Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to correct max
relu
"deconv"
relu
"deconv"
Gradients
Guided backprop


Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to correct max
relu
"deconv"
relu
"deconv"
Gradients
Guided backprop

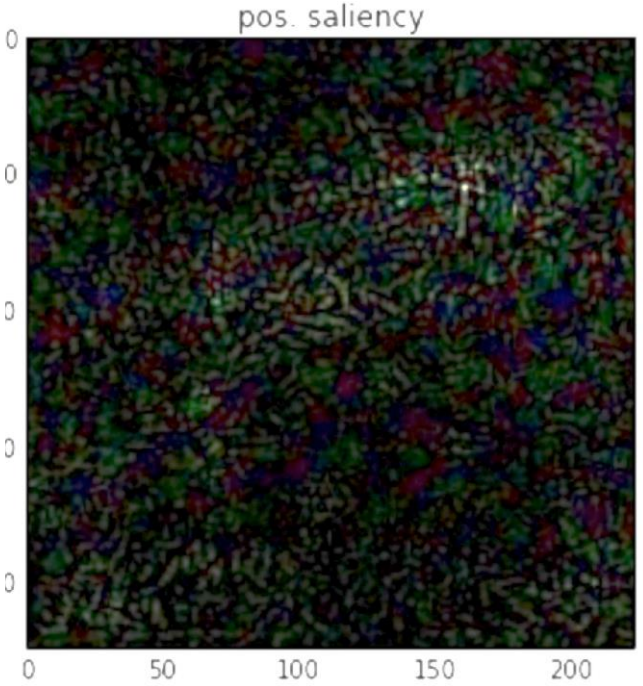

Forward pass
max
relu
conv
relu
conv
Deconv
"switch" to correct max
relu
"deconv"
relu
"deconv"
Gradients
Guided backprop



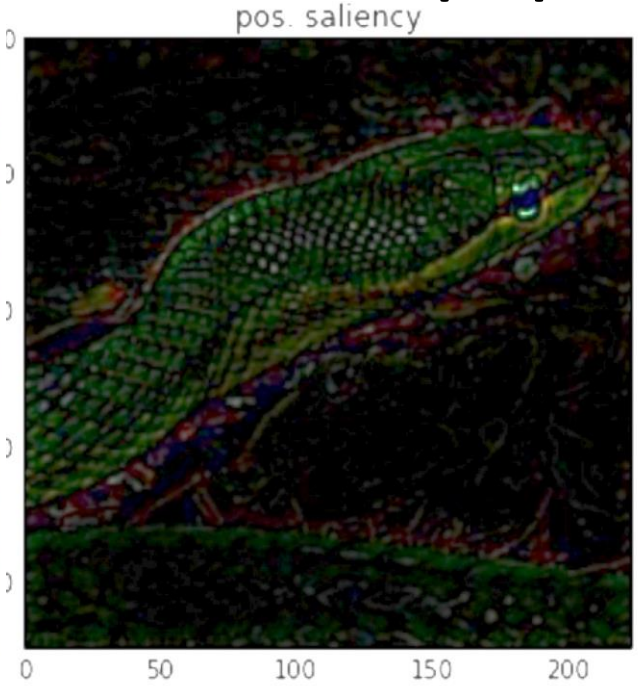
Deconv



Gradients



Guided Backprop



Summary of differences

ReLU(x) = max(0, x)

Forward-pass

1	-1	5
2	-5	-7
-3	2	4

ReLU →

1	0	5
2	0	0
0	2	4

Summary of differences

ReLU(x) = max(0, x)

Forward-pass

1	-1	5
2	-5	-7
-3	2	4

ReLU →

1	0	5
2	0	0
0	2	4

Red = input < 0
Yellow = gradient < 0

← Gradients

-2	3	-1
6	-3	1
2	-1	3

← Deconv

-2	3	-1
6	-3	1
2	-1	3

← Guided backprop

-2	3	-1
6	-3	1
2	-1	3

Summary of differences

ReLU(x) = max(0, x)

Forward-pass

1	-1	5
2	-5	-7
-3	2	4

ReLU →

1	0	5
2	0	0
0	2	4

Red = input < 0
Yellow = gradient < 0

Zero-out gradients through inactive ReLUs

-2	0	-1
6	0	0
0	-1	3

←

-2	3	-1
6	-3	1
2	-1	3

Gradients

←

-2	3	-1
6	-3	1
2	-1	3

Deconv

←

-2	3	-1
6	-3	1
2	-1	3

Guided backprop

Summary of differences

ReLU(x) = max(0, x)

Forward-pass

1	-1	5
2	-5	-7
-3	2	4

ReLU →

1	0	5
2	0	0
0	2	4

Red = input < 0
Yellow = gradient < 0

Zero-out gradients through inactive ReLUs

-2	0	-1
6	0	0
0	-1	3

←

-2	3	-1
6	-3	1
2	-1	3

Gradients

Zero-out negative gradients & ignore ReLU

0	3	0
6	0	1
2	0	3

←

-2	3	-1
6	-3	1
2	-1	3

Deconv

←

-2	3	-1
6	-3	1
2	-1	3

Guided backprop

Summary of differences

ReLU(x) = max(0, x)

Forward-pass

1	-1	5
2	-5	-7
-3	2	4

ReLU →

1	0	5
2	0	0
0	2	4

Red = input < 0
Yellow = gradient < 0

Zero-out gradients through inactive ReLUs

-2	0	-1
6	0	0
0	-1	3

←

-2	3	-1
6	-3	1
2	-1	3

Gradients

Zero-out negative gradients & ignore ReLU

0	3	0
6	0	1
2	0	3

←

-2	3	-1
6	-3	1
2	-1	3

Deconv

(Combined)
Zero-out inactive ReLUs
Zero-out negative gradients

0	0	0
6	0	0
0	0	3

←

-2	3	-1
6	-3	1
2	-1	3

Guided backprop

Summary of differences

ReLU(x) = max(0, x)

Forward-pass

1	-1	5
2	-5	-7
-3	2	4

ReLU →

1	0	5
2	0	0
0	2	4

Red = input < 0
Yellow = gradient < 0

Zero-out gradients through inactive ReLUs

-2	0	-1
6	0	0
0	-1	3

←

-2	3	-1
6	-3	1
2	-1	3

Gradients

Zero-out negative gradients & ignore ReLU

0	3	0
6	0	1
2	0	3

←

-2	3	-1
6	-3	1
2	-1	3

Deconv

(Combined)
Zero-out inactive ReLUs
Zero-out negative gradients

0	0	0
6	0	0
0	0	3

←

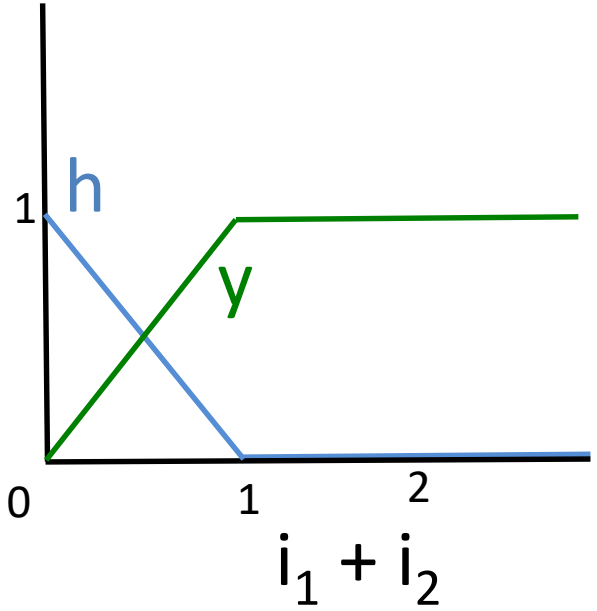
-2	3	-1
6	-3	1
2	-1	3

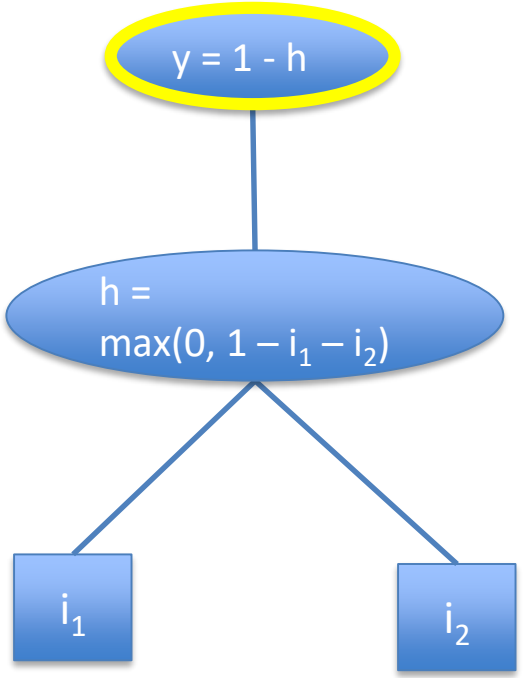
Guided backprop

No consideration of nonlinearities other than ReLU
No special handling of biases

Example failure-mode 1: saturation

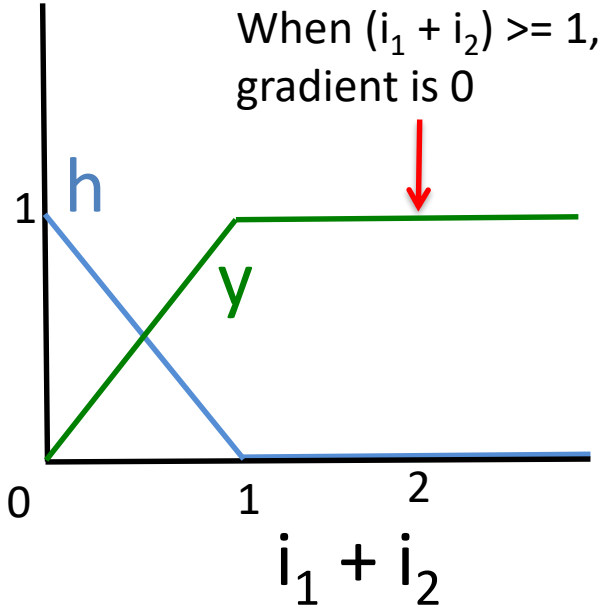
$y = (i_1 + i_2)$ when $(i_1 + i_2) < 1$
$= 1$ when $(i_1 + i_2) >= 1$

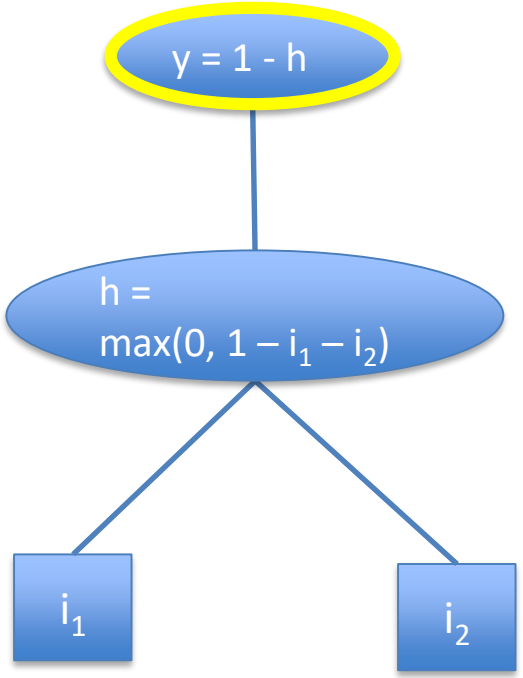




Example failure-mode 1: saturation

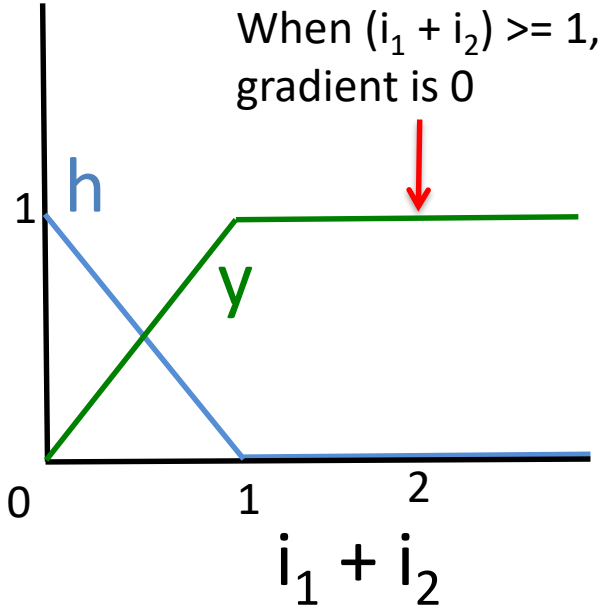
$y = (i_1 + i_2)$ when $(i_1 + i_2) < 1$
$= 1$ when $(i_1 + i_2) >= 1$

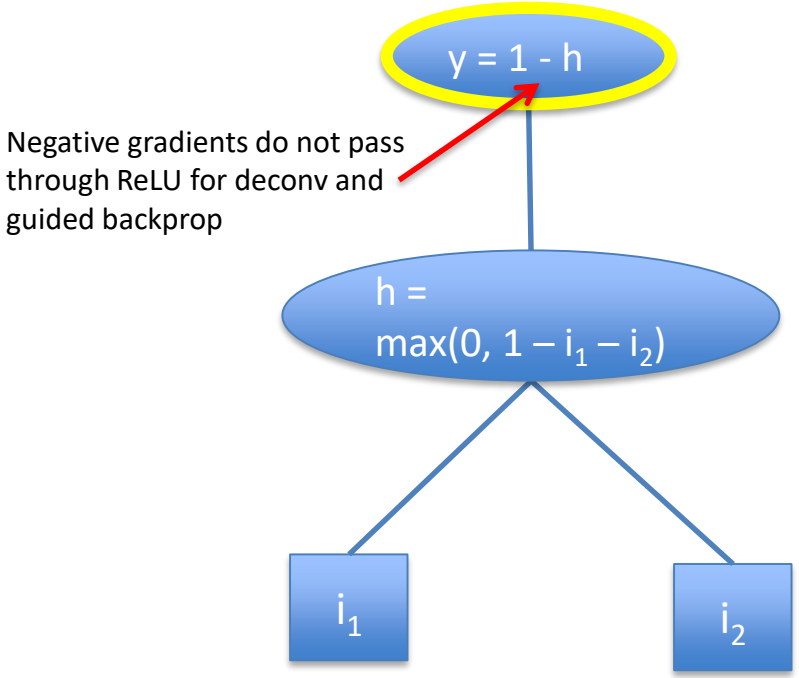




Example failure-mode 1: saturation

$y = (i_1 + i_2)$ when $(i_1 + i_2) < 1$
$= 1$ when $(i_1 + i_2) >= 1$



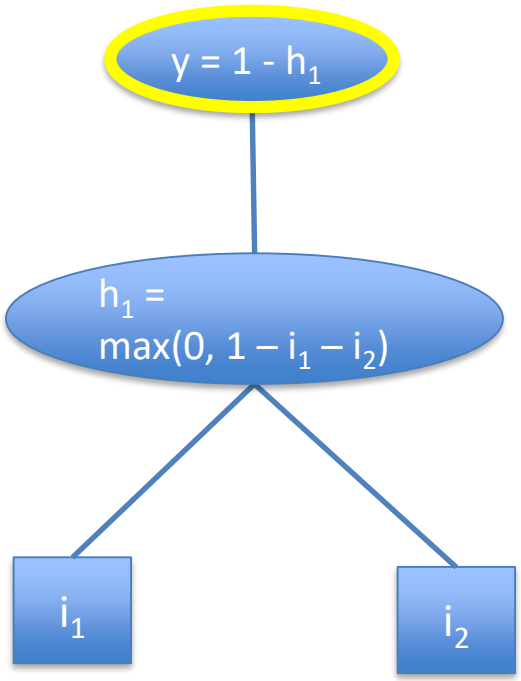


Idea 4: DeepLIFT

- Moves away from gradients, relying on "difference from reference"
- Solves saturation problem and other problems
- Equivalent to gradient*input when biases in the network are all zero

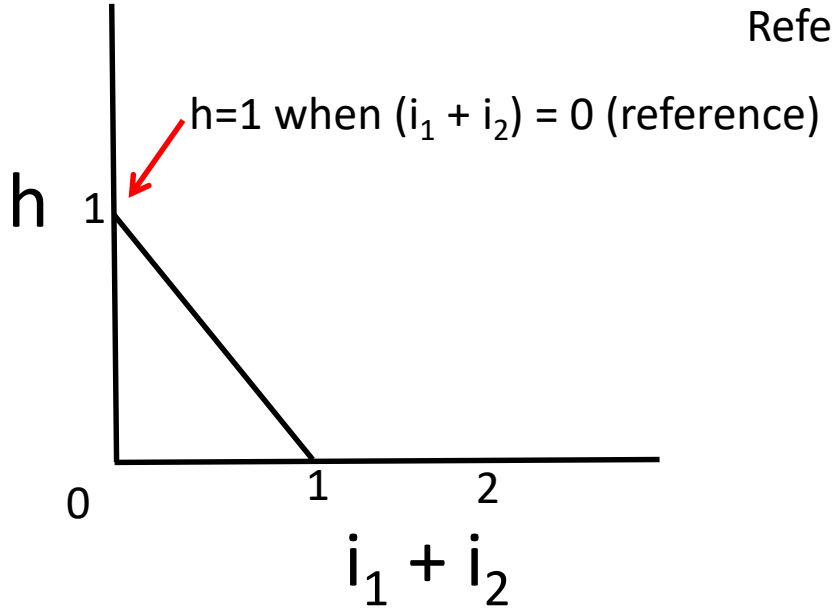
The DeepLIFT solution: difference from reference

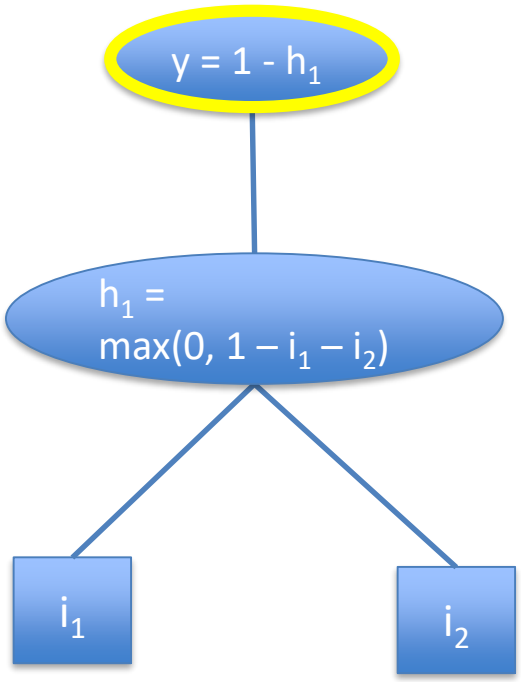
$y = (i_1 + i_2)$ when $(i_1 + i_2) < 1$
$= 1$ when $(i_1 + i_2) >= 1$
Reference: $i_1=0$ & $i_2=0$



The DeepLIFT solution: difference from reference

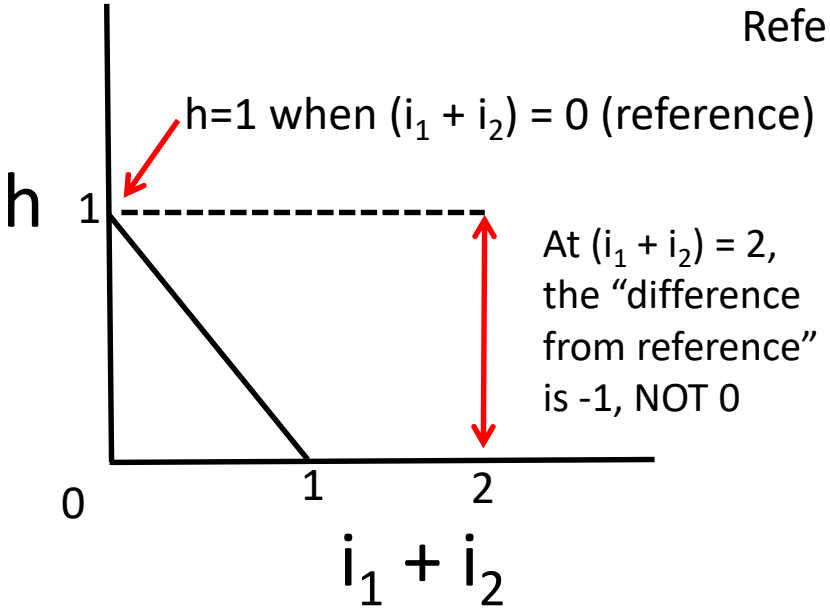
$y = (i_1 + i_2)$ when $(i_1 + i_2) < 1$
$= 1$ when $(i_1 + i_2) >= 1$
Reference: $i_1=0$ & $i_2=0$

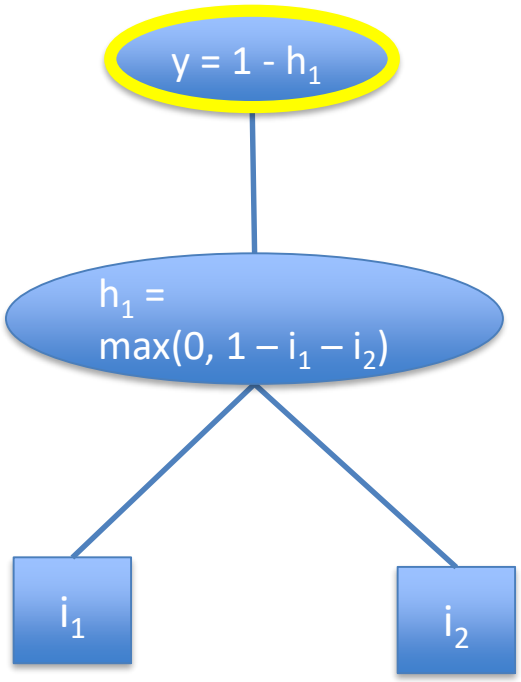




The DeepLIFT solution: difference from reference

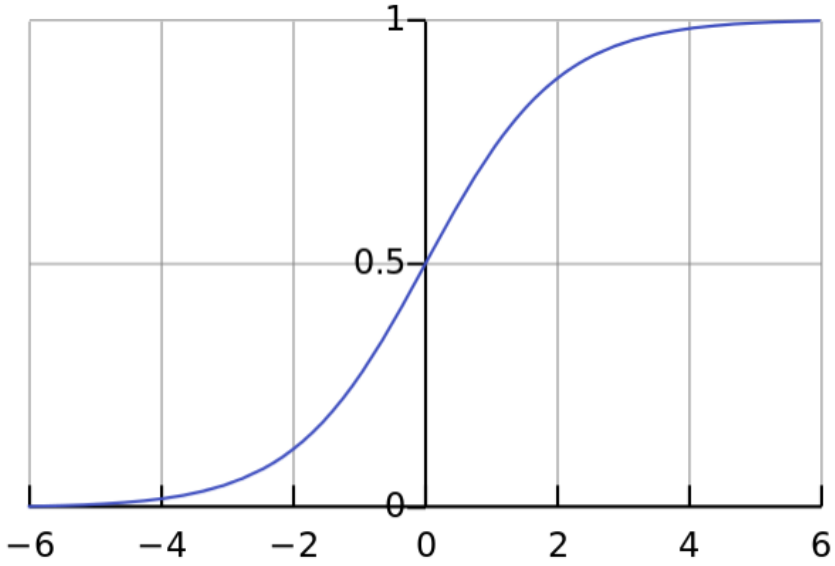
$y = (i_1 + i_2)$ when $(i_1 + i_2) < 1$
$= 1$ when $(i_1 + i_2) >= 1$
Reference: $i_1=0$ & $i_2=0$





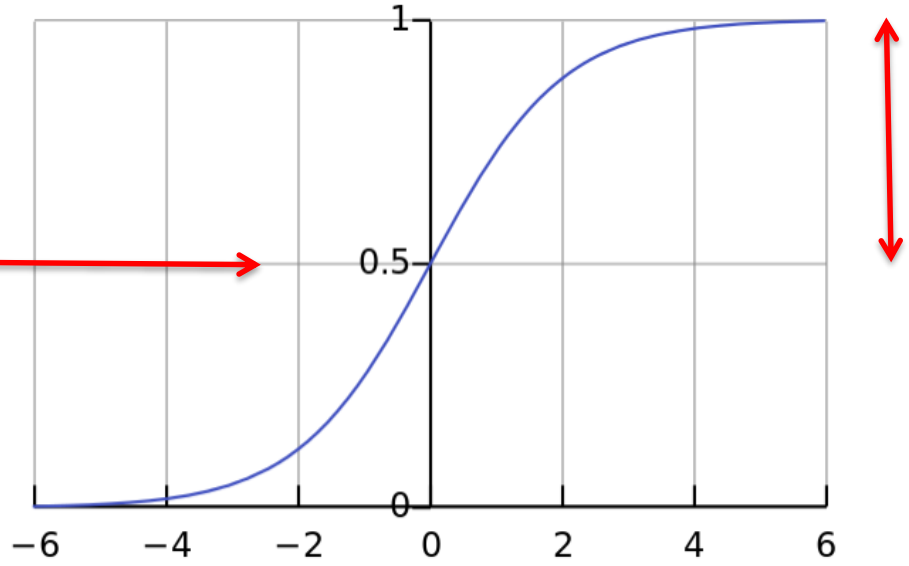
DeepLIFT generalizes to all activations

DeepLIFT generalizes to all activations



DeepLIFT generalizes to all activations

Sigmoid is 0.5 when input is 0



“difference from reference” is +0.5 when inputs is >> 0 (assuming reference input of 0)

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

Let $A^0_{x_i}$ represent the “reference” activation of x_i

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

Let $A^0_{x_i}$ represent the “reference” activation of x_i

$$A^0_y = f(A^0_{x_1}, A^0_{x_2}, \ldots, A^0_{x_i})$$

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

Let $A^0_{x_i}$ represent the “reference” activation of x_i

$$A^0_y = f(A^0_{x_1}, A^0_{x_2}, \ldots, A^0_{x_i})$$

$$\Delta_{x_i} = A_{x_i} - A^0_{x_i}$$

$$\Delta_y = A_y - A^0_y$$

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

Let $A^0_{x_i}$ represent the “reference” activation of x_i

$$A^0_y = f(A^0_{x_1}, A^0_{x_2}, \ldots, A^0_{x_i})$$

$$\Delta_{x_i} = A_{x_i} - A^0_{x_i}$$

$$\Delta_y = A_y - A^0_y$$

$C_{x_i y}$ is the portion of Δ_y attributed to Δ_{x_i}

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

Let $A^0_{x_i}$ represent the “reference” activation of x_i

$$A^0_y = f(A^0_{x_1}, A^0_{x_2}, \ldots, A^0_{x_i})$$

$$\Delta_{x_i} = A_{x_i} - A^0_{x_i}$$

$$\Delta_y = A_y - A^0_y$$

$C_{x_i y}$ is the portion of Δ_y attributed to Δ_{x_i}

Partial derivatives	DeepLIFT multipliers

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

Let $A^0_{x_i}$ represent the “reference” activation of x_i

$$A^0_y = f(A^0_{x_1}, A^0_{x_2}, \ldots, A^0_{x_i})$$

$$\Delta_{x_i} = A_{x_i} - A^0_{x_i}$$

$$\Delta_y = A_y - A^0_y$$

$C_{x_i y}$ is the portion of Δ_y attributed to Δ_{x_i}

Partial derivatives	DeepLIFT multipliers
$\frac{\partial_y}{\partial_{x_i}} = \frac{\text{change in } y \text{ due to } x_i}{\text{change in } x_i}$	

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

Let $A^0_{x_i}$ represent the “reference” activation of x_i

$$A^0_y = f(A^0_{x_1}, A^0_{x_2}, \ldots, A^0_{x_i})$$

$$\Delta_{x_i} = A_{x_i} - A^0_{x_i}$$

$$\Delta_y = A_y - A^0_y$$

$C_{x_i y}$ is the portion of Δ_y attributed to Δ_{x_i}

Partial derivatives	DeepLIFT multipliers
$\frac{\partial_y}{\partial_{x_i}} = \frac{\text{change in } y \text{ due to } x_i}{\text{change in } x_i}$	$m_{x_i y} = \frac{C_{x_i y}}{\Delta x_i}$

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

Let $A^0_{x_i}$ represent the “reference” activation of x_i

$$A^0_y = f(A^0_{x_1}, A^0_{x_2}, \ldots, A^0_{x_i})$$

$$\Delta_{x_i} = A_{x_i} - A^0_{x_i}$$

$$\Delta_y = A_y - A^0_y$$

$C_{x_i y}$ is the portion of Δ_y attributed to Δ_{x_i}

Partial derivatives	DeepLIFT multipliers
$\frac{\partial_y}{\partial_{x_i}} = \frac{\text{change in } y \text{ due to } x_i}{\text{change in } x_i}$	$m_{x_i y} = \frac{C_{x_i y}}{\Delta x_i}$

$$\frac{\partial_y}{\partial_w} = \sum_i \frac{\partial_{x_i}}{\partial_w} \frac{\partial_y}{\partial_{x_i}}$$

(chain rule)

“Multipliers”: alternative to gradients

$$A_y = f(A_{x_1}, A_{x_2}, \ldots, A_{x_i})$$

Let $A^0_{x_i}$ represent the “reference” activation of x_i

$$A^0_y = f(A^0_{x_1}, A^0_{x_2}, \ldots, A^0_{x_i})$$

$$\Delta_{x_i} = A_{x_i} - A^0_{x_i}$$

$$\Delta_y = A_y - A^0_y$$

$C_{x_i y}$ is the portion of Δ_y attributed to Δ_{x_i}

Partial derivatives	DeepLIFT multipliers
$\frac{\partial_y}{\partial_{x_i}} = \frac{\text{change in } y \text{ due to } x_i}{\text{change in } x_i}$	$m_{x_i y} = \frac{C_{x_i y}}{\Delta x_i}$
$\frac{\partial y}{\partial w} = \sum_i \frac{\partial x_i}{\partial w} \frac{\partial y}{\partial x_i}$ (chain rule)	$m_{wy} = \sum_i m_{wx_i} m_{x_i y}$

Multiplier definitions

Function	Gradient	DeepLIFT multiplier
Linear $A_y = b + \sum_i w_i A_{x_i}$	$\frac{\partial_{A_y}}{\partial_{A_{x_i}}} = w_i$	
ReLU $A_y = \max(0, A_x)$		

Multiplier definitions

Function	Gradient	DeepLIFT multiplier
Linear $A_y = b + \sum_i w_i A_{x_i}$	$\frac{\partial_{A_y}}{\partial_{A_{x_i}}} = w_i$	$C_{x_i y} = w_i \Delta_{x_i}$ $m_{x_i y} = \frac{C_{x_i y}}{\Delta_{x_i}} = w_i$
ReLU $A_y = \max(0, A_x)$		

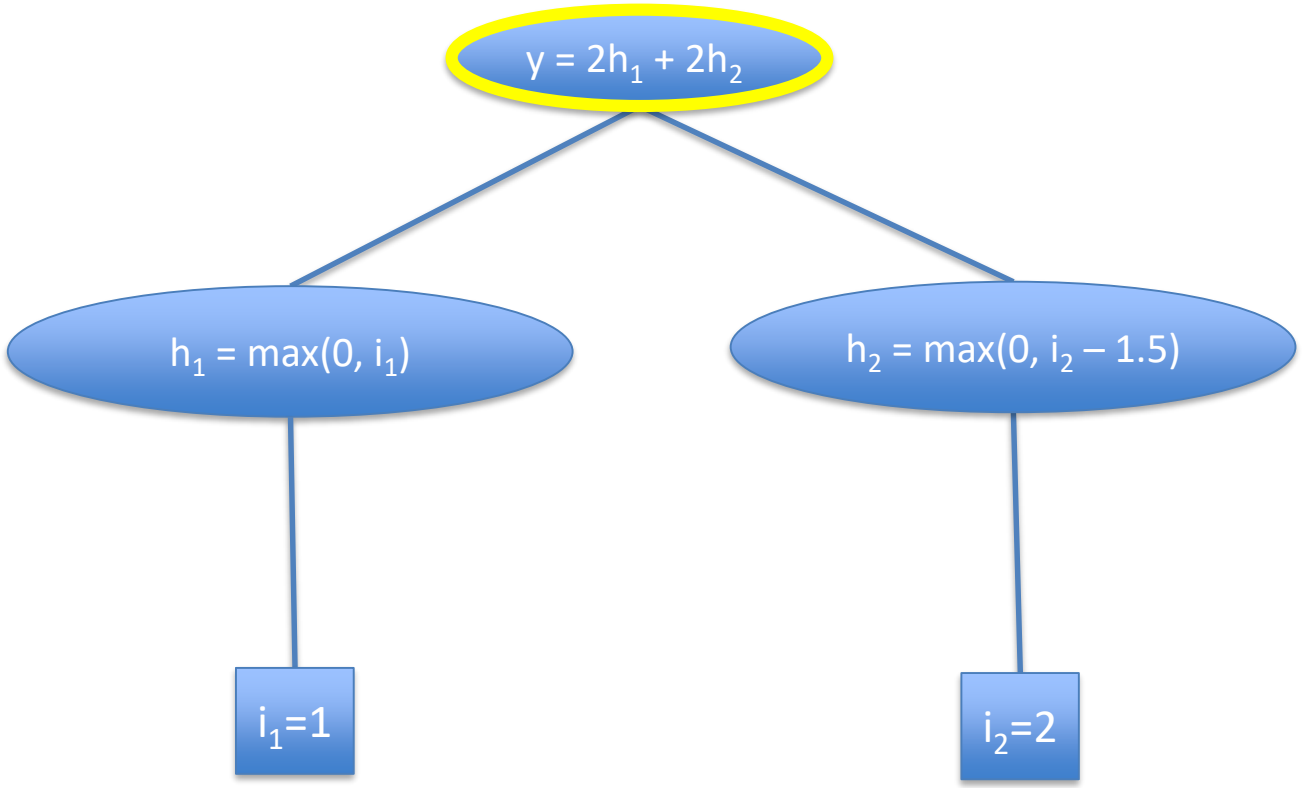
Multiplier definitions

Function	Gradient	DeepLIFT multiplier
Linear $A_y = b + \sum_i w_i A_{x_i}$	$\frac{\partial_{A_y}}{\partial_{A_{x_i}}} = w_i$	$C_{x_i y} = w_i \Delta_{x_i}$ $m_{x_i y} = \frac{C_{x_i y}}{\Delta_{x_i}} = w_i$
ReLU $A_y = \max(0, A_x)$	$\frac{\partial_{A_y}}{\partial_{A_x}} = 1 \text{ if } x > 0$ $= 0 \text{ if } x \leq 0$	

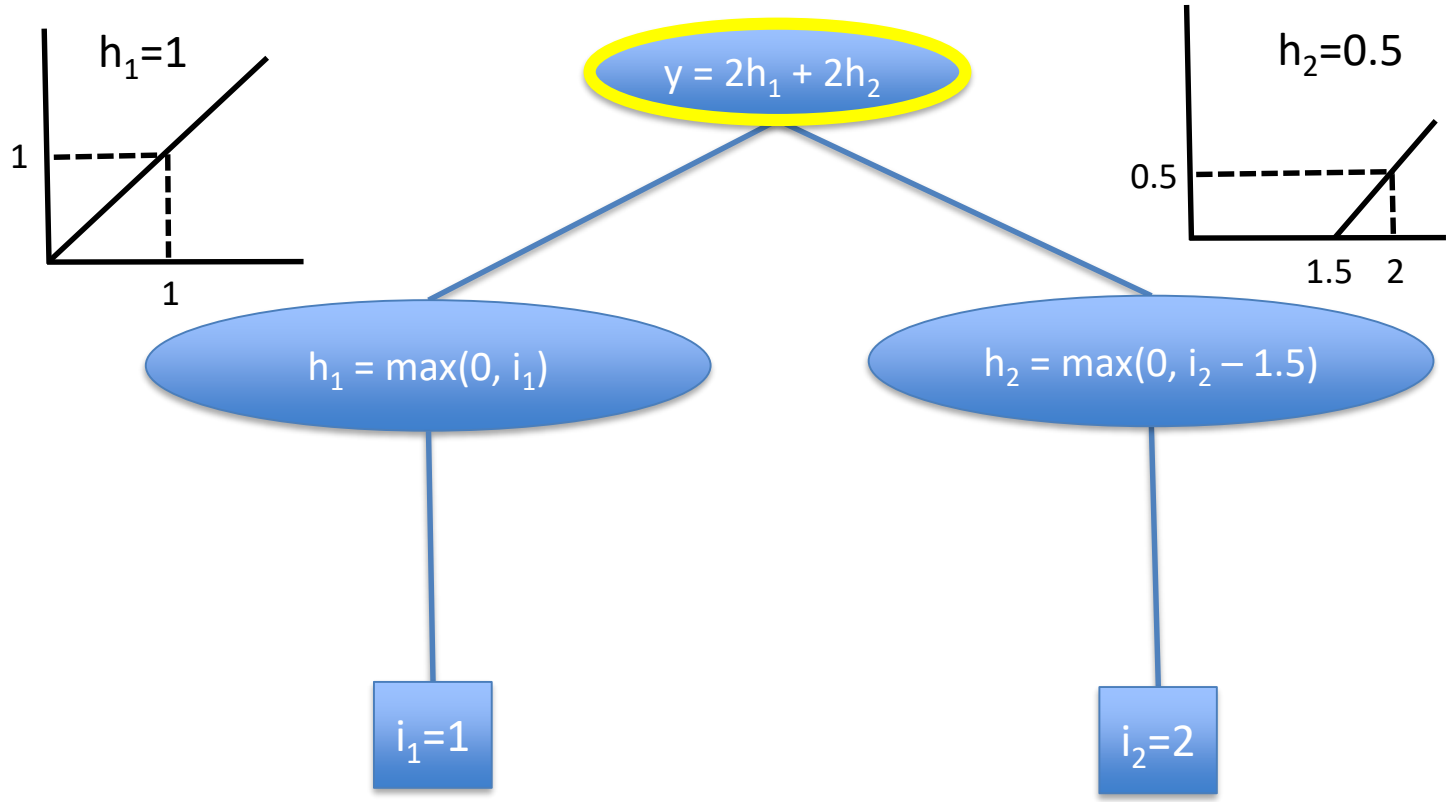
Multiplier definitions

Function	Gradient	DeepLIFT multiplier
Linear $A_y = b + \sum_i w_i A_{x_i}$	$\frac{\partial_{A_y}}{\partial_{A_{x_i}}} = w_i$	$C_{x_i y} = w_i \Delta_{x_i}$ $m_{x_i y} = \frac{C_{x_i y}}{\Delta_{x_i}} = w_i$
ReLU $A_y = \max(0, A_x)$	$\frac{\partial_{A_y}}{\partial_{A_x}} = 1 \text{ if } x > 0$ $= 0 \text{ if } x \leq 0$	$C_{xy} = \Delta_y$ $m_{xy} = \frac{\Delta_y}{\Delta_x}$

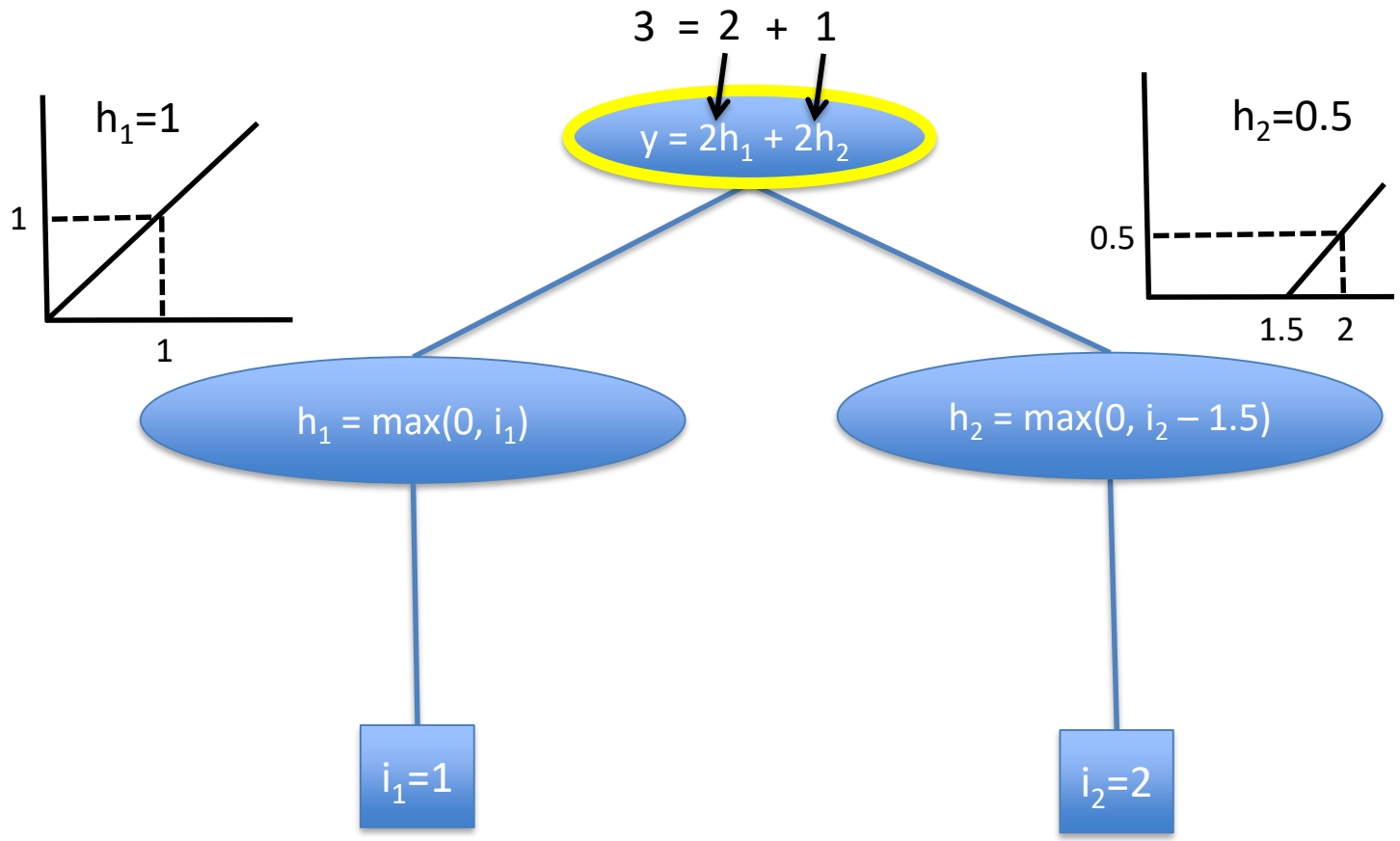
Example failure-mode 2: thresholds



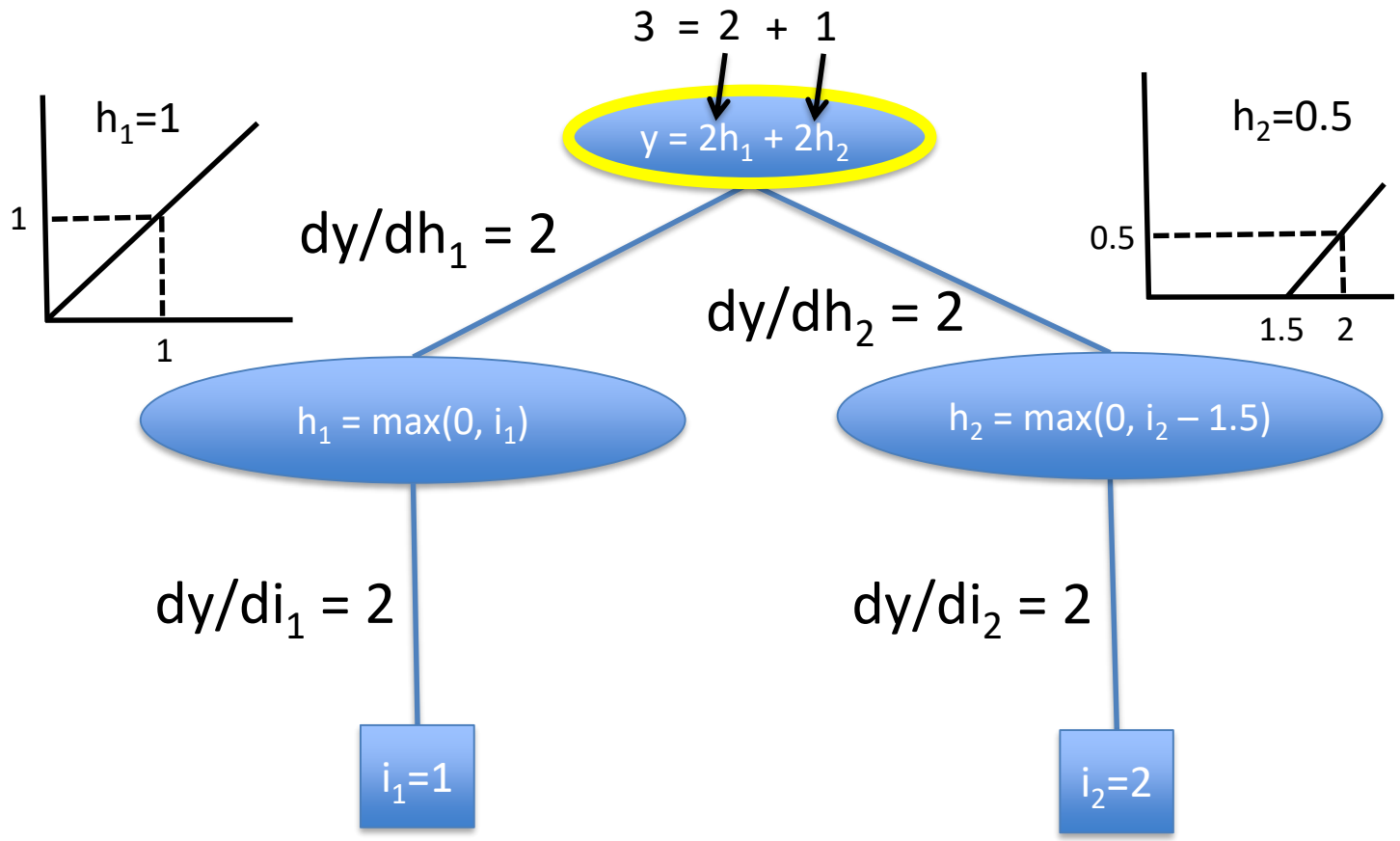
Example failure-mode 2: thresholds



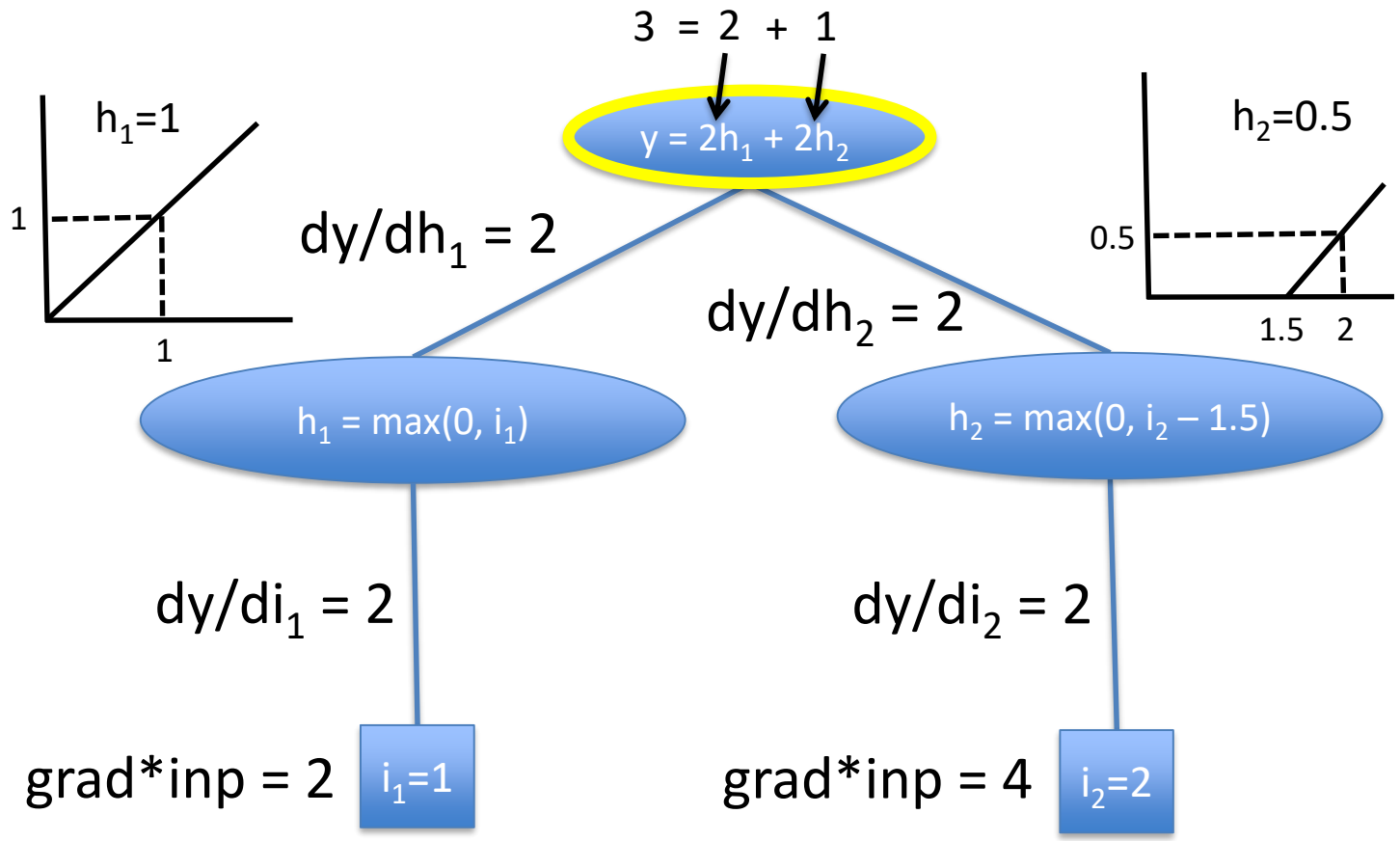
Example failure-mode 2: thresholds



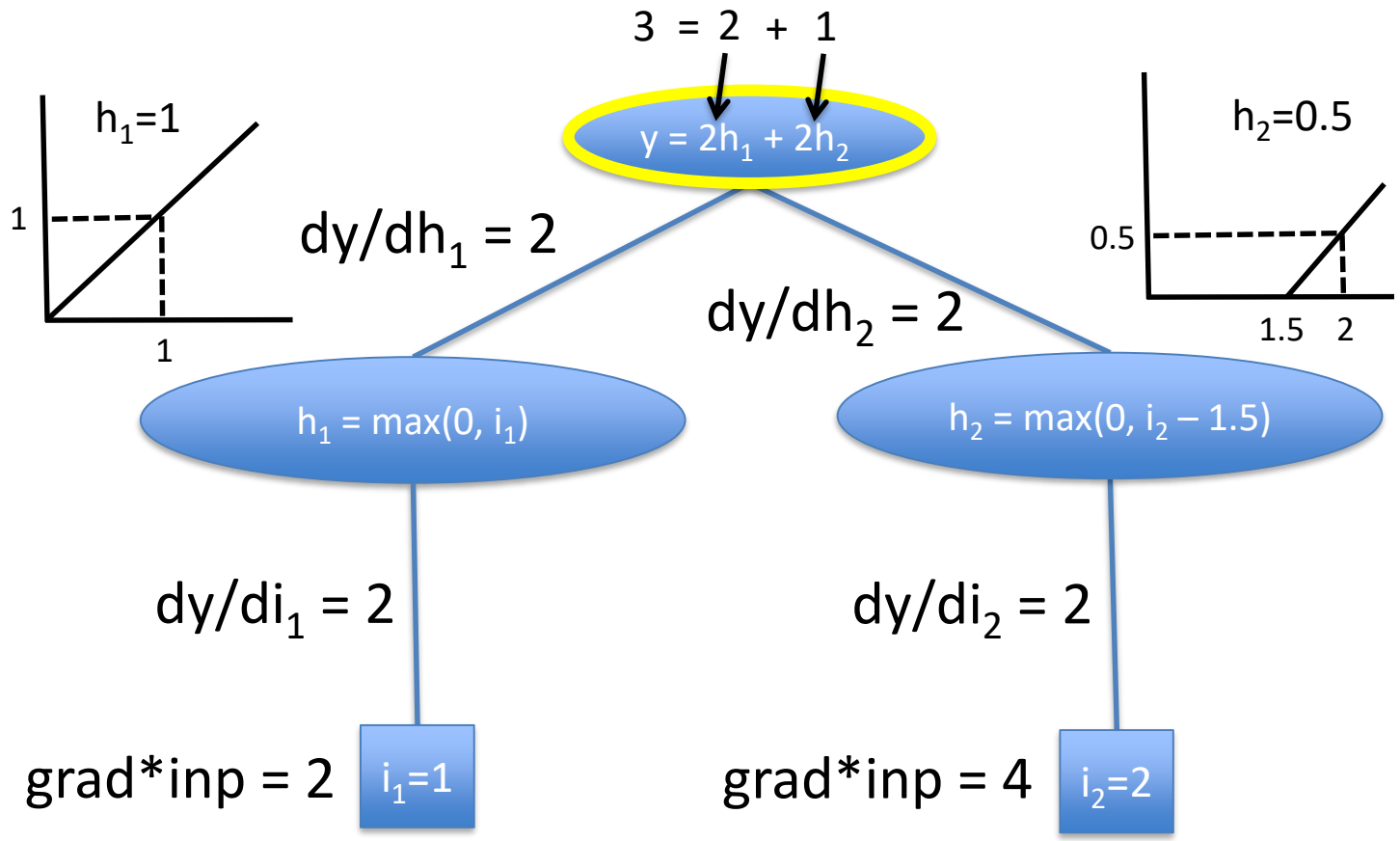
Example failure-mode 2: thresholds



Example failure-mode 2: thresholds

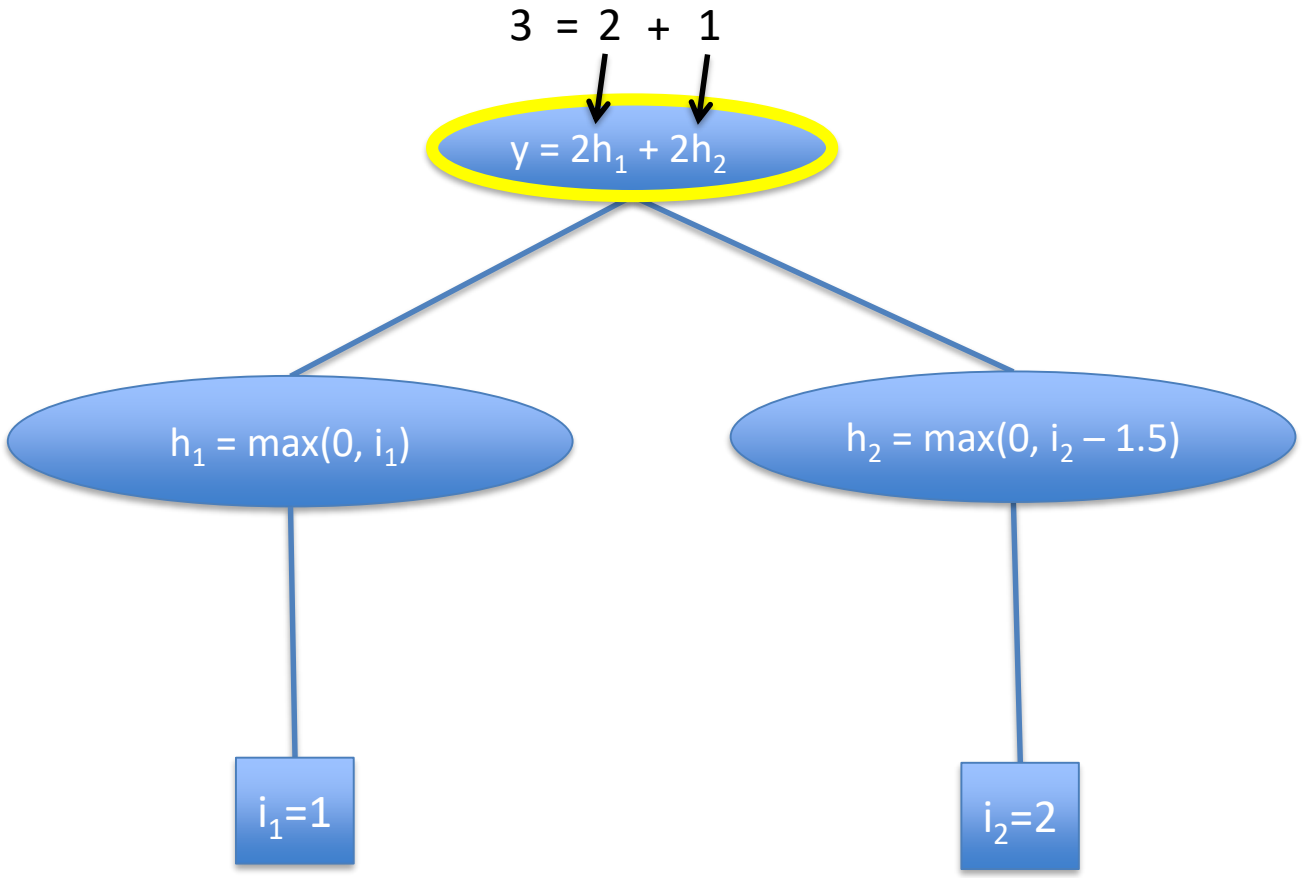


Example failure-mode 2: thresholds

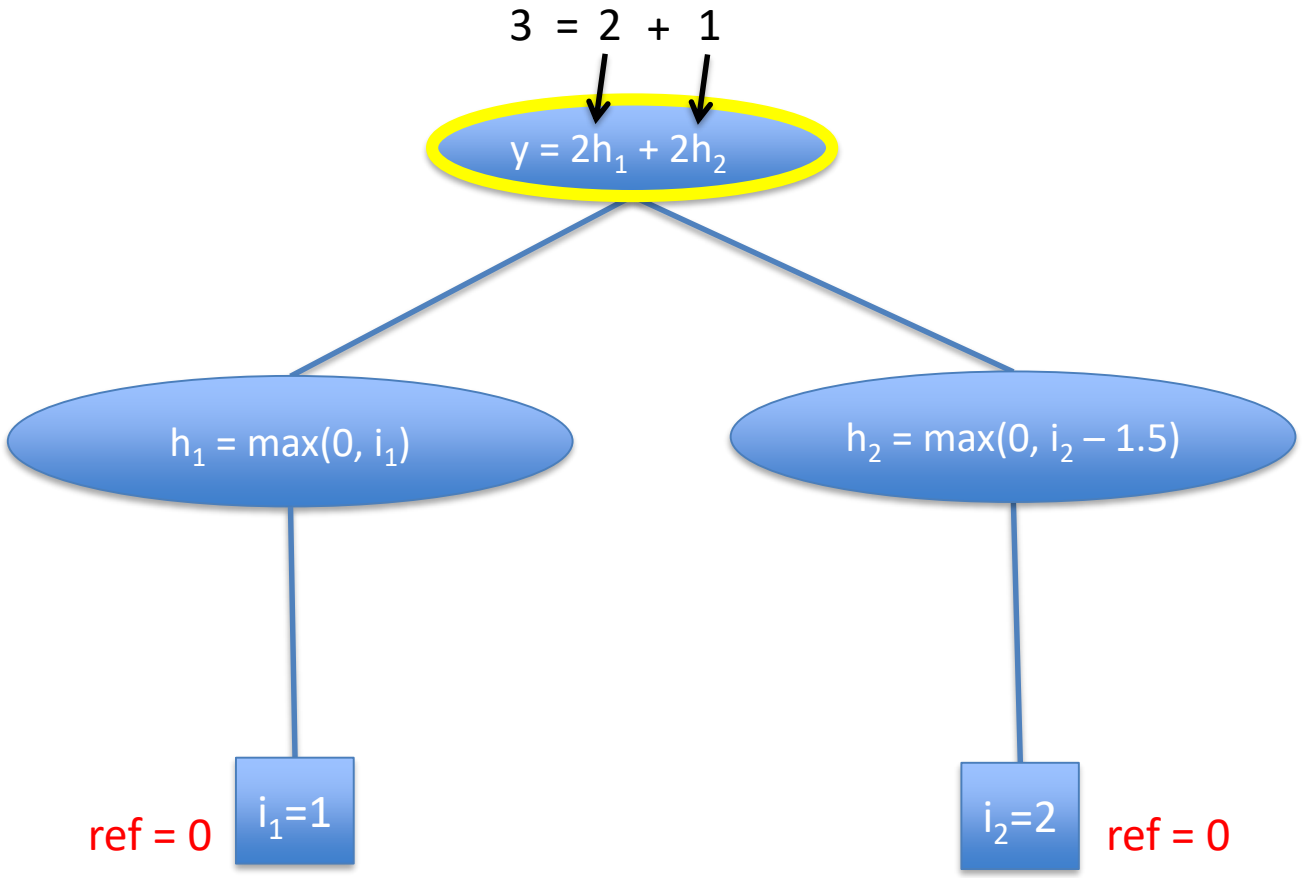


Contributions from i_1 & i_2 are 2 and 1 respectively, but gradient*Δinput is 2 and 4 respectively

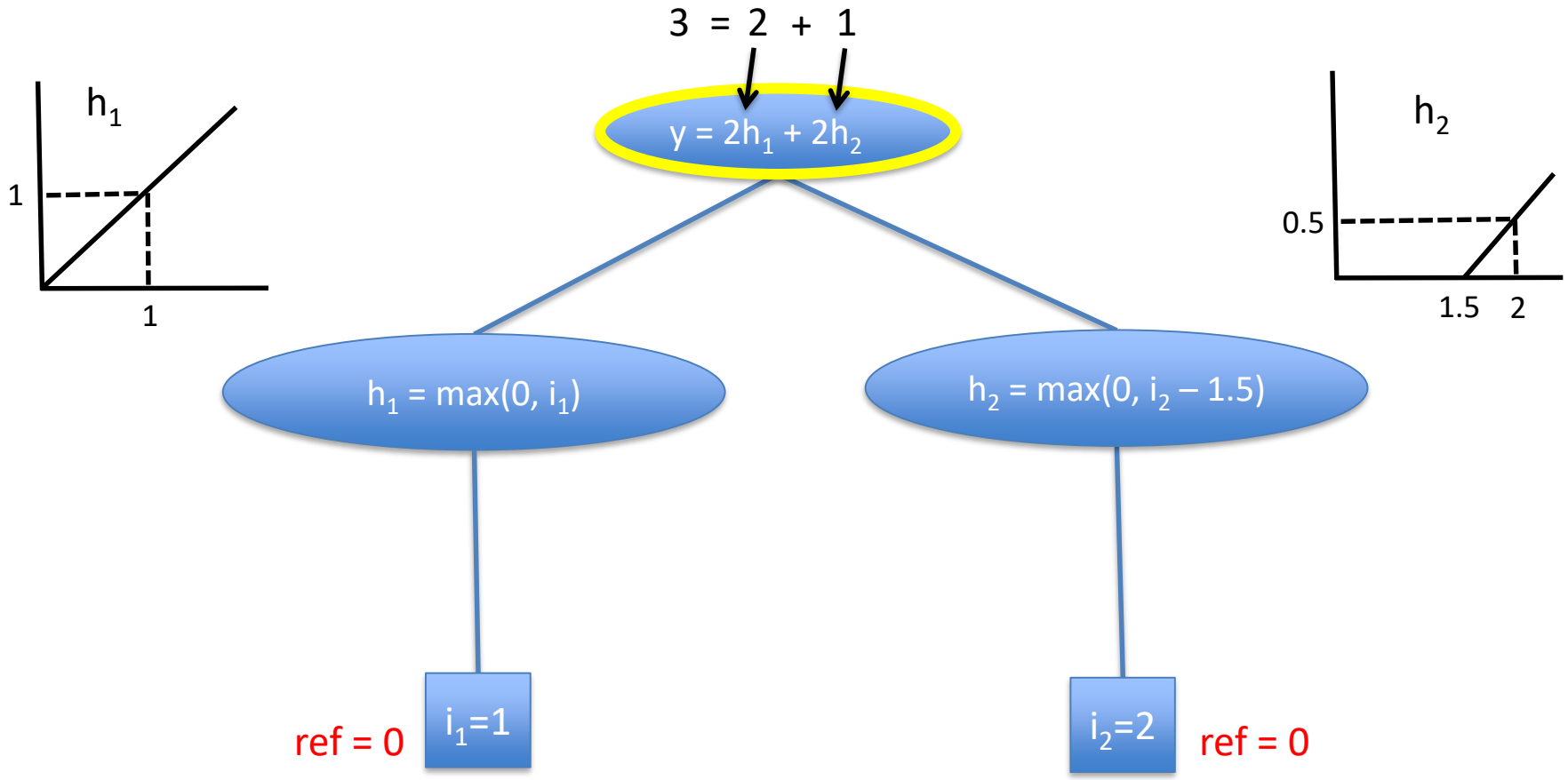
Solution with DeepLIFT



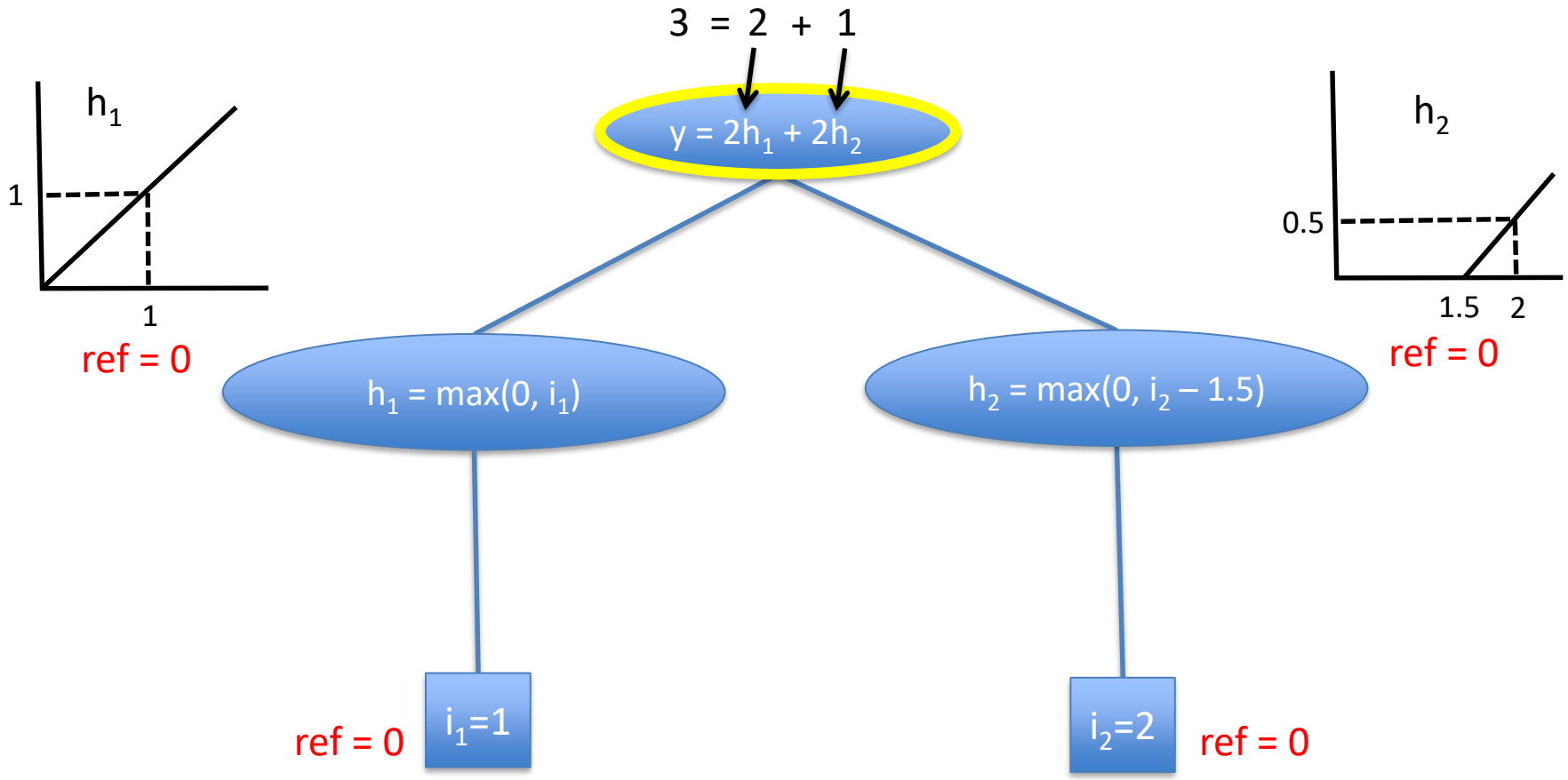
Solution with DeepLIFT



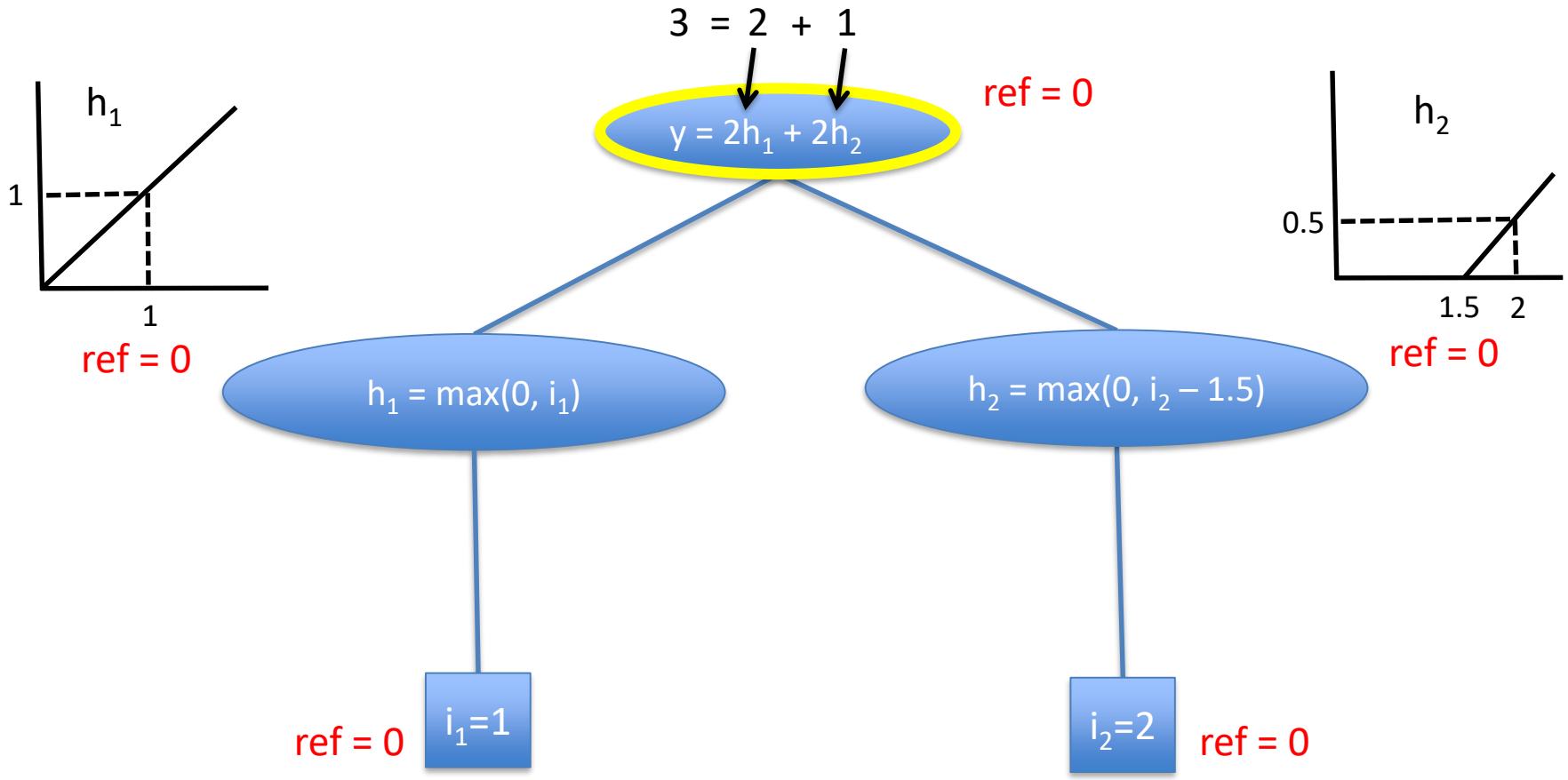
Solution with DeepLIFT



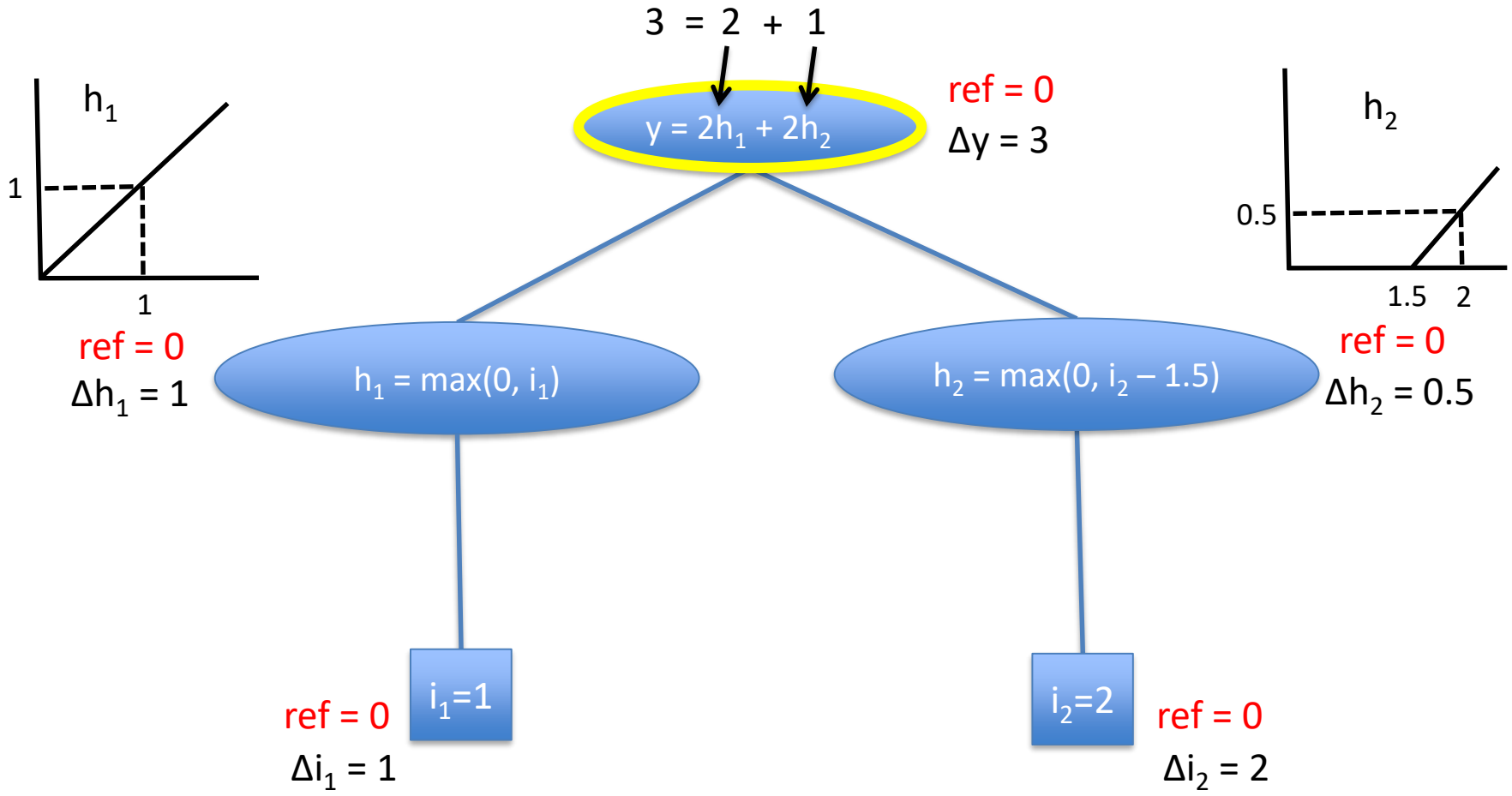
Solution with DeepLIFT



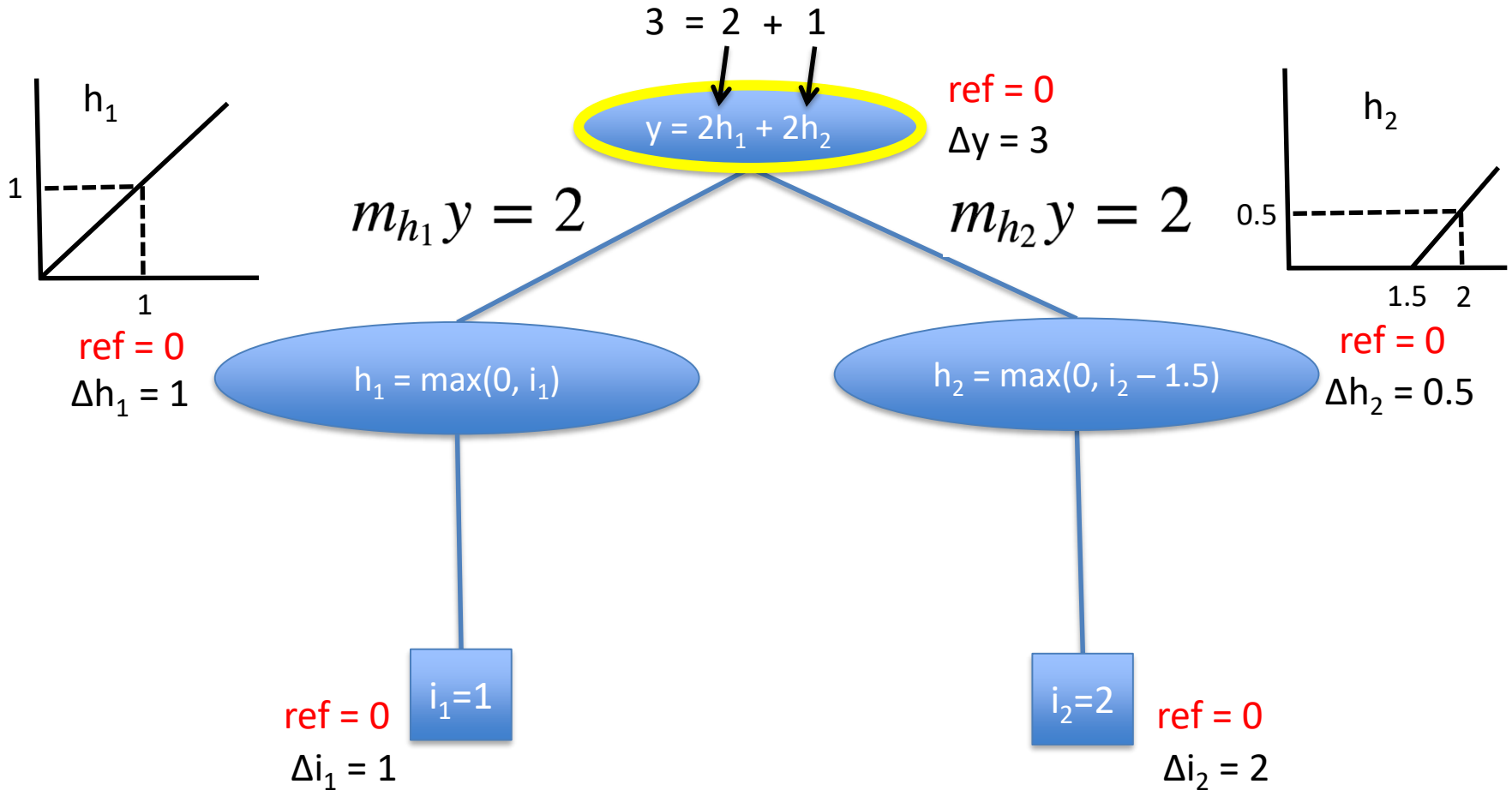
Solution with DeepLIFT



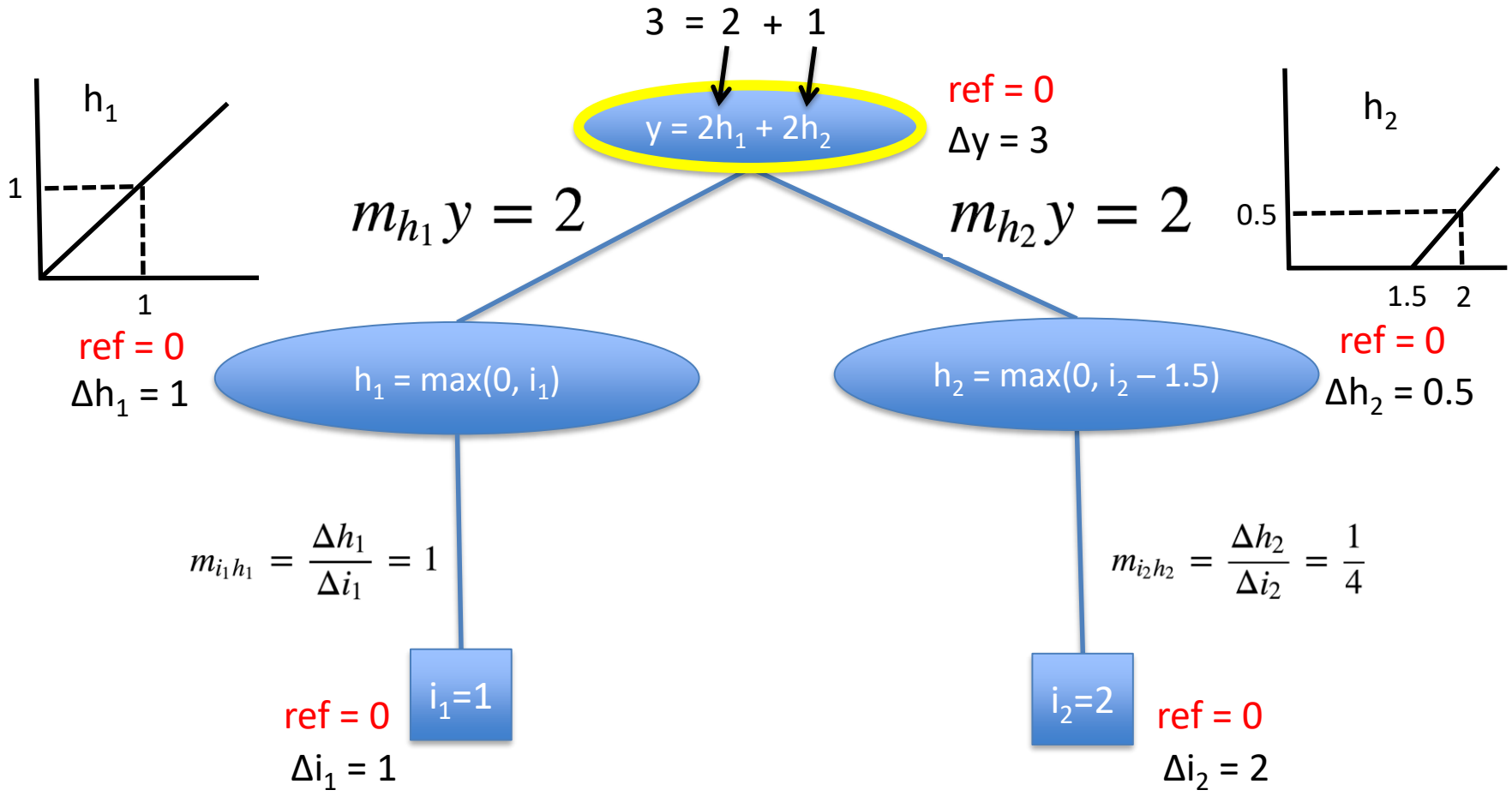
Solution with DeepLIFT



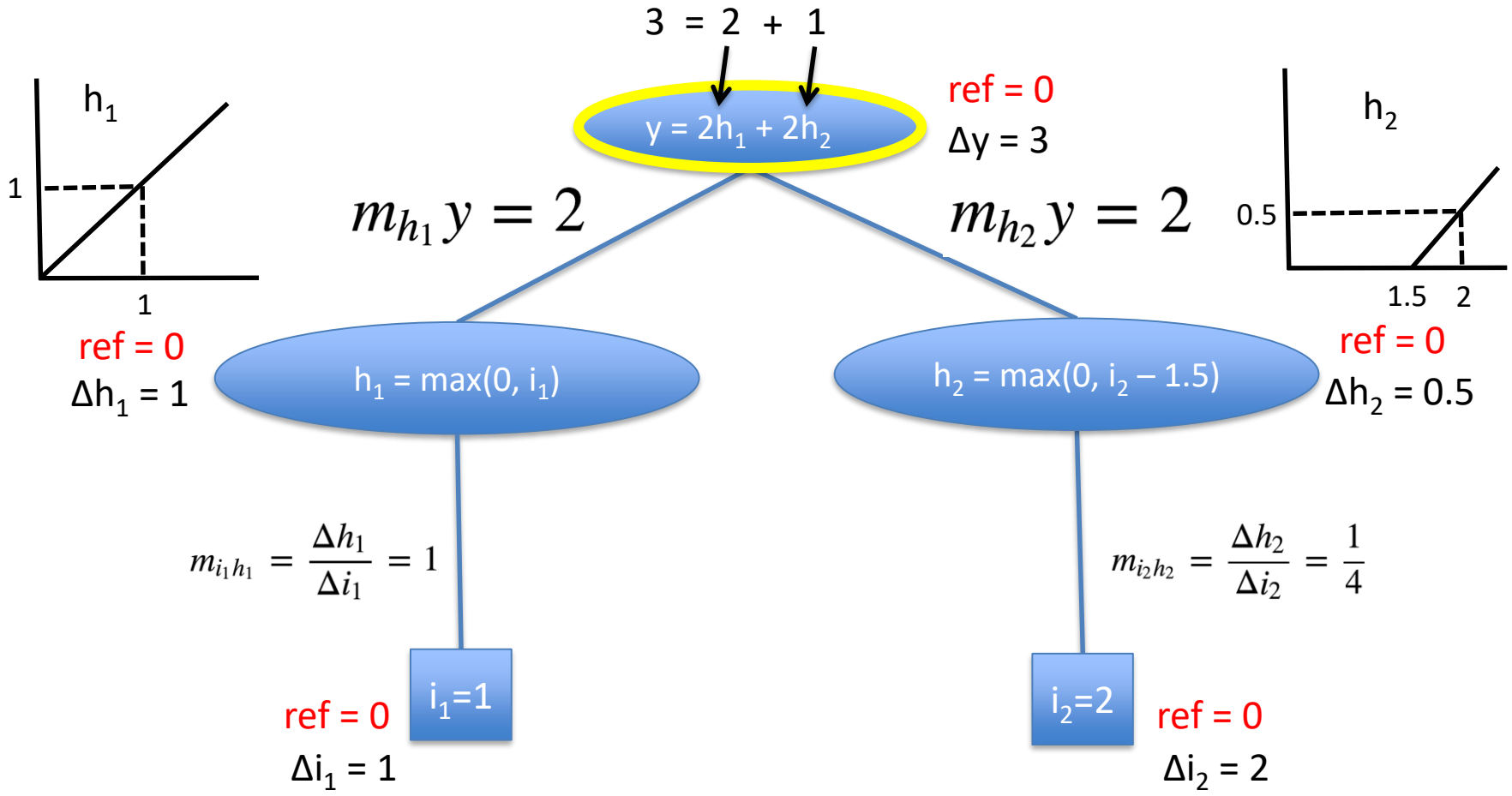
Solution with DeepLIFT



Solution with DeepLIFT



Solution with DeepLIFT



$$m_{h_1 y} = 2$$

$$m_{h_2 y} = 2$$

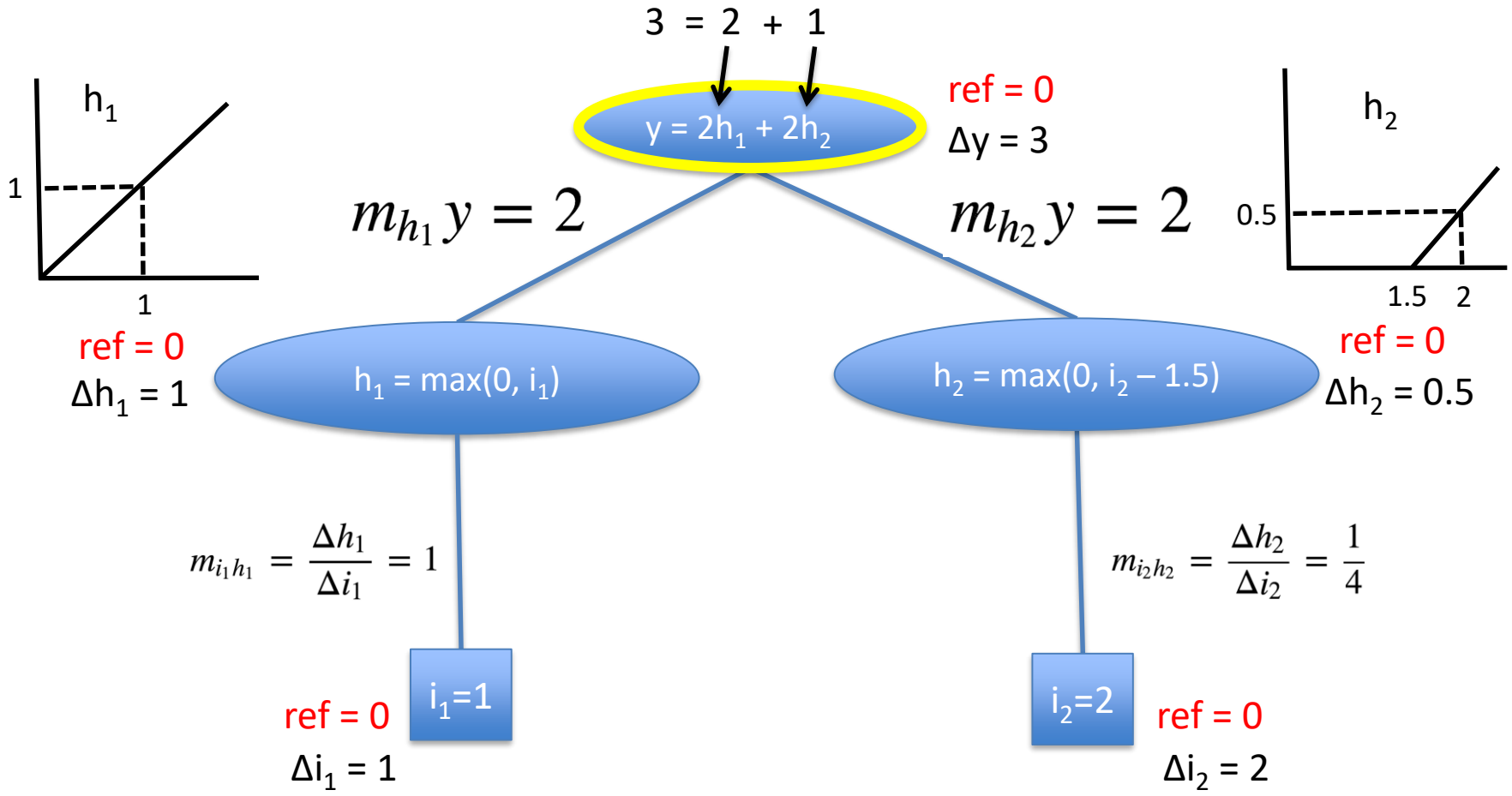
$$m_{i_1 h_1} = \frac{\Delta h_1}{\Delta i_1} = 1$$

$$m_{i_2 h_2} = \frac{\Delta h_2}{\Delta i_2} = \frac{1}{4}$$

$$m_{i_1 y} = m_{i_1 h_1} m_{h_1 y} = 2$$

$$m_{i_2 y} = m_{i_2 h_2} m_{h_2 y} = 0.5$$

Solution with DeepLIFT



$$m_{h_1} y = 2$$

$$m_{h_2} y = 2$$

$$m_{i_1 h_1} = \frac{\Delta h_1}{\Delta i_1} = 1$$

$$m_{i_2 h_2} = \frac{\Delta h_2}{\Delta i_2} = \frac{1}{4}$$

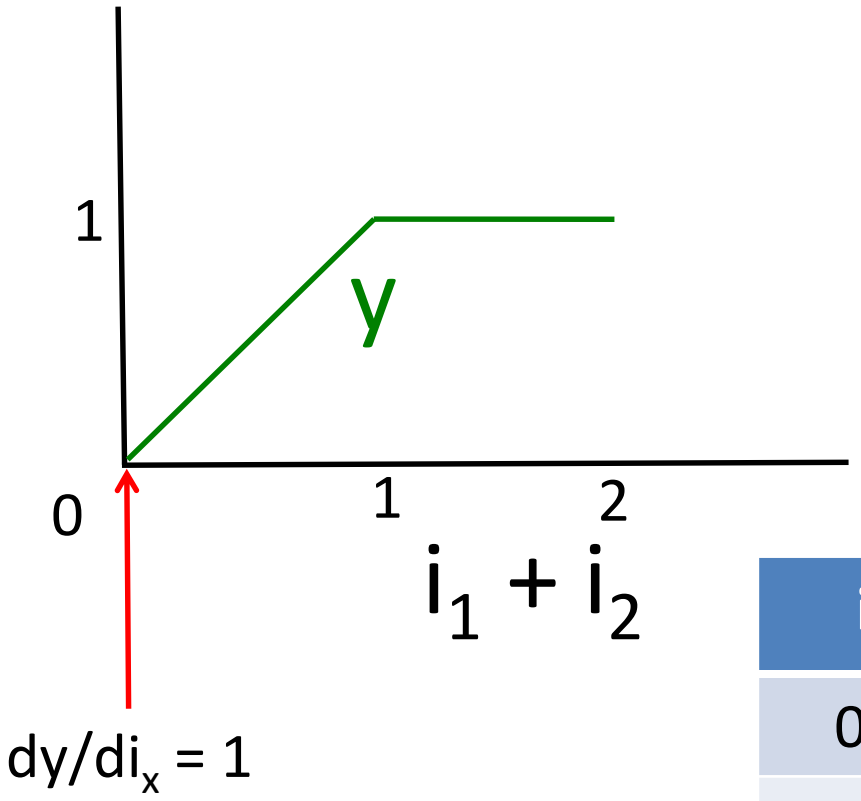
$$m_{i_1 y} = m_{i_1 h_1} m_{h_1 y} = 2$$
$$C_{i_1 y} = m_{i_1 y} \Delta_{i_1} = 2$$

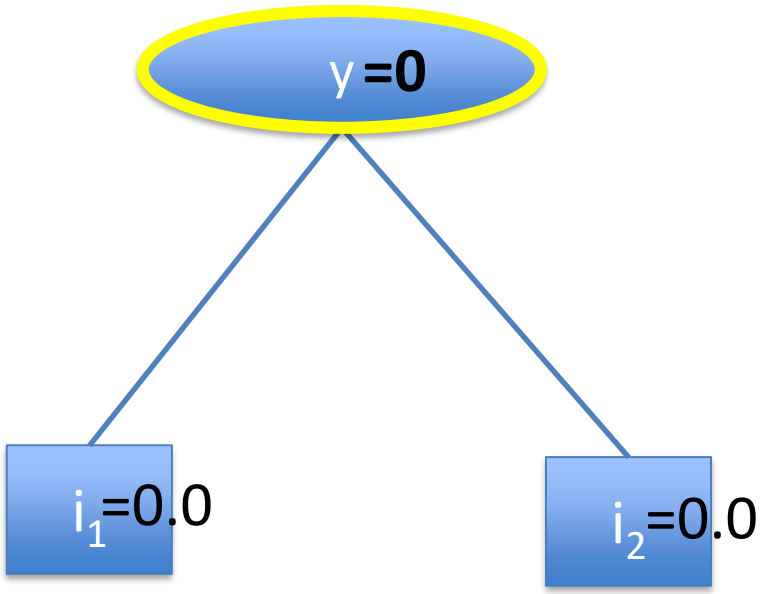
$$m_{i_2 y} = m_{i_2 h_2} m_{h_2 y} = 0.5$$
$$C_{i_2 y} = m_{i_2 y} \Delta_{i_2} = 1$$

Idea 4: Integrated Gradients

- Moves away from gradients, relying on "difference from reference"
- Solves saturation problem and other problems
- Equivalent to gradient*input when biases in the network are all zero

Integrated Gradients: Another reference-based approach

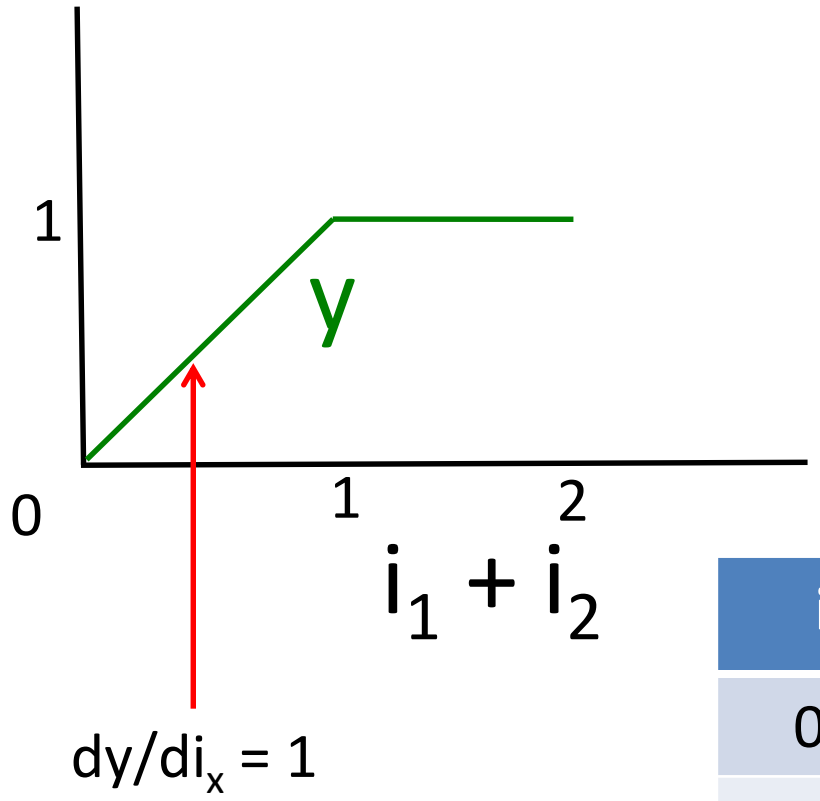


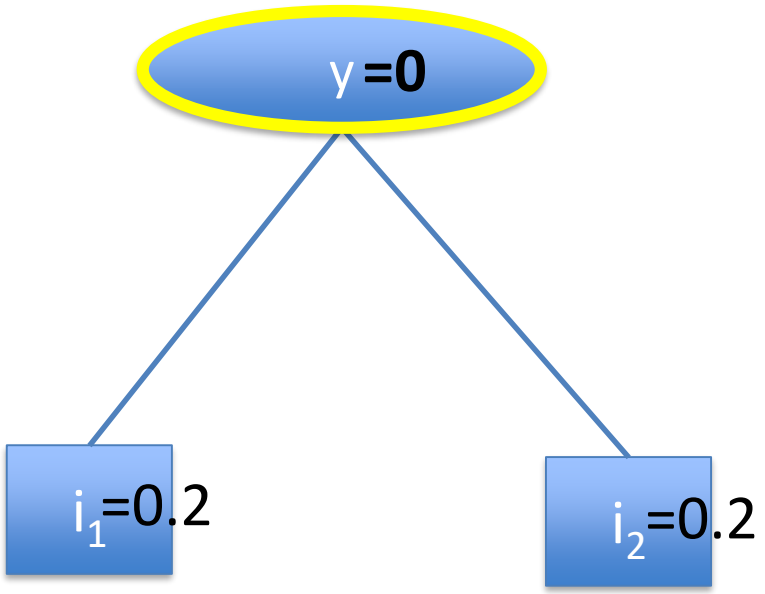


i_1	i_2	dy/di_x
0.0	0.0	1

i_1	i_2	dy/di_x

Integrated Gradients: Another reference-based approach

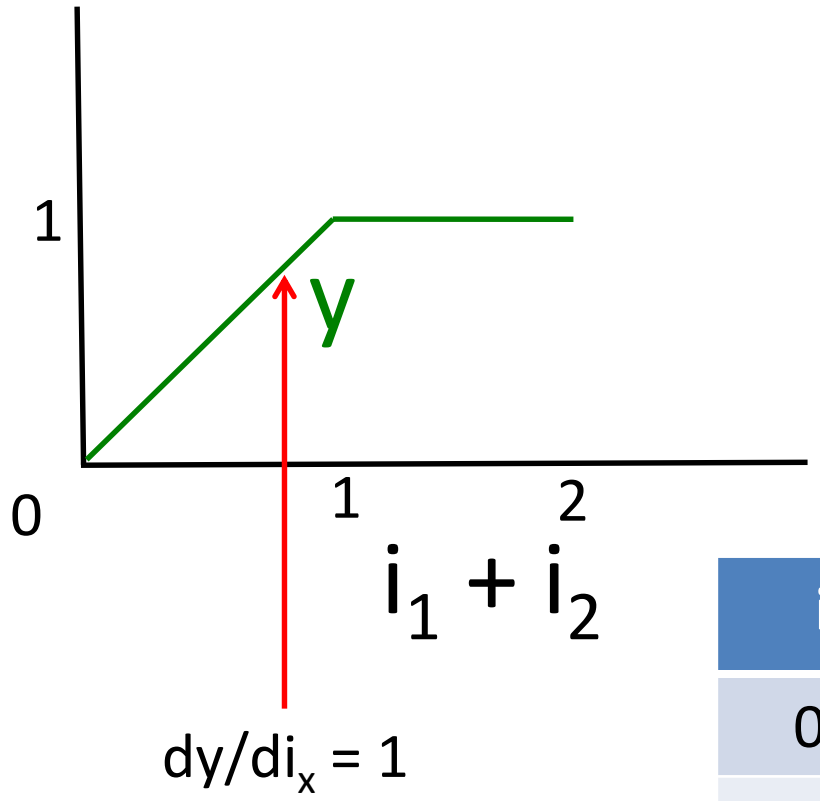


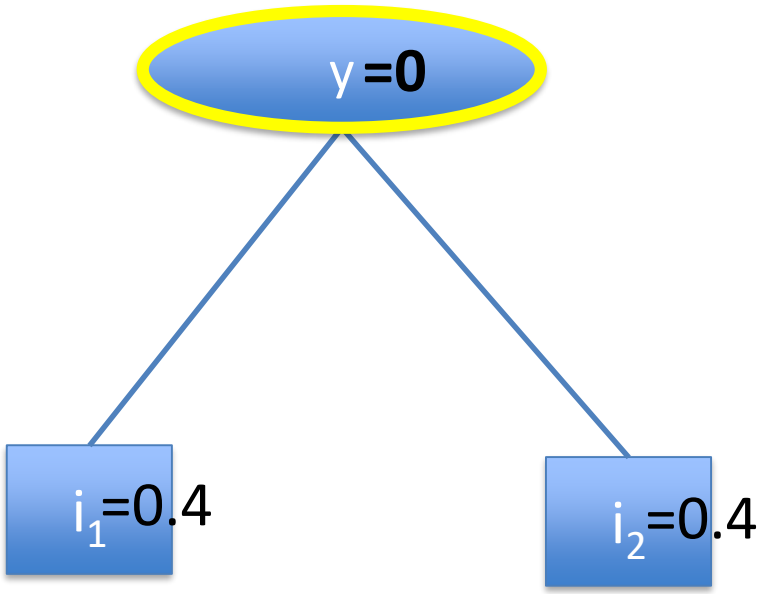


i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1

i_1	i_2	dy/di_x

Integrated Gradients: Another reference-based approach

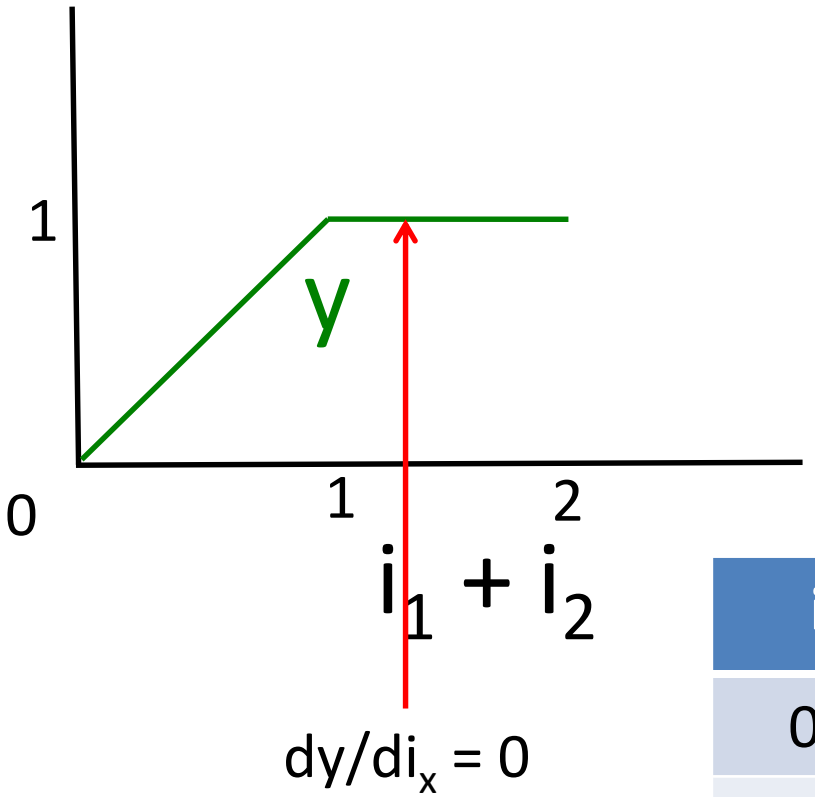


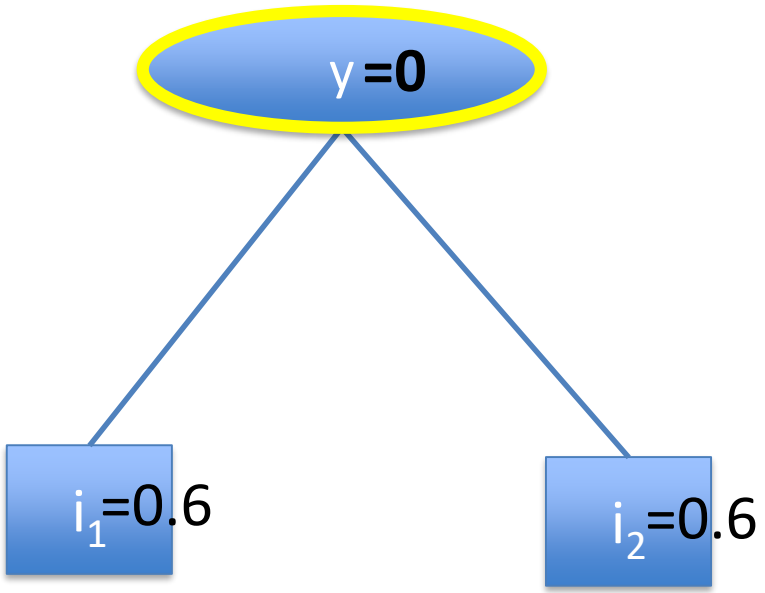


i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1
0.4	0.4	1

i_1	i_2	dy/di_x

Integrated Gradients: Another reference-based approach

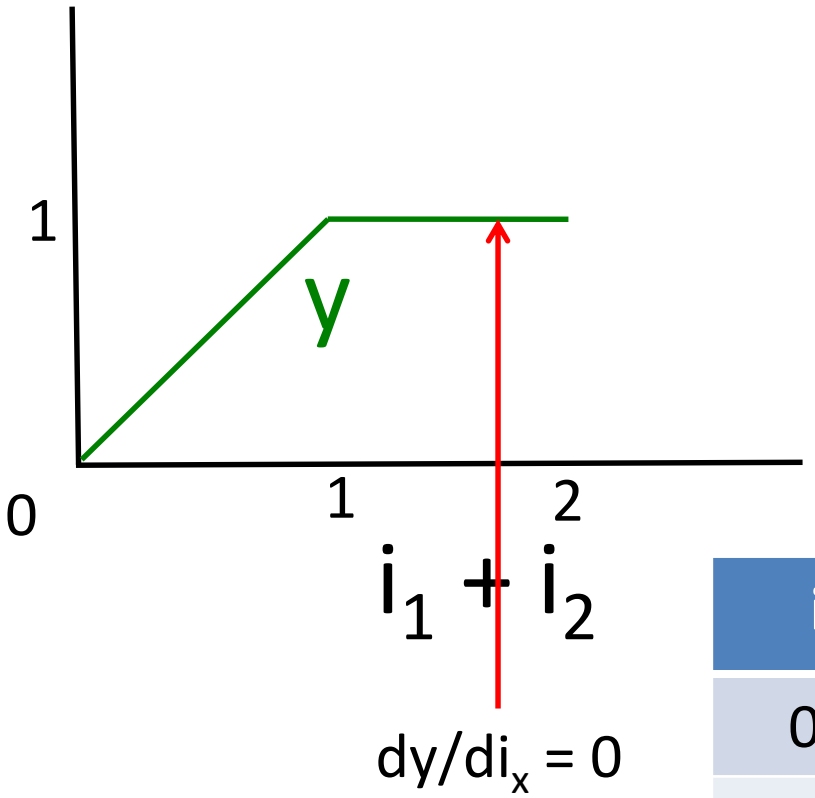


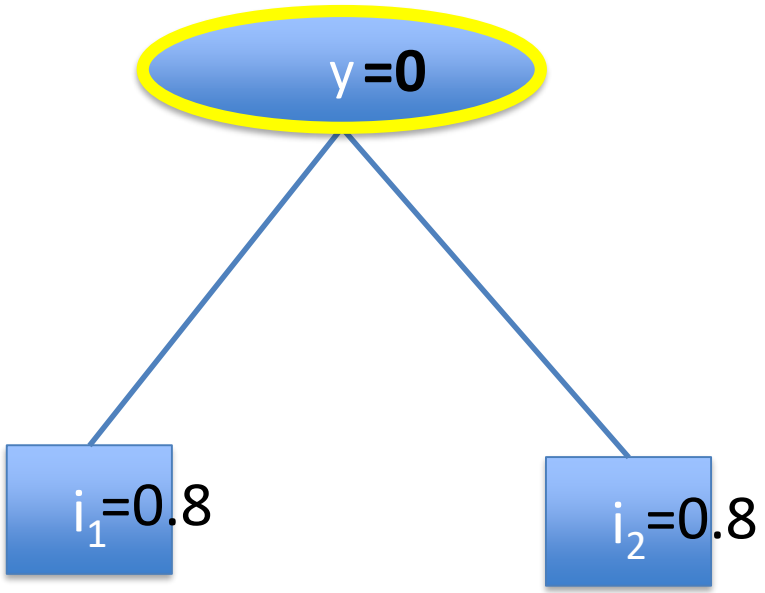


i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1
0.4	0.4	1

i_1	i_2	dy/di_x
0.6	0.6	0

Integrated Gradients: Another reference-based approach

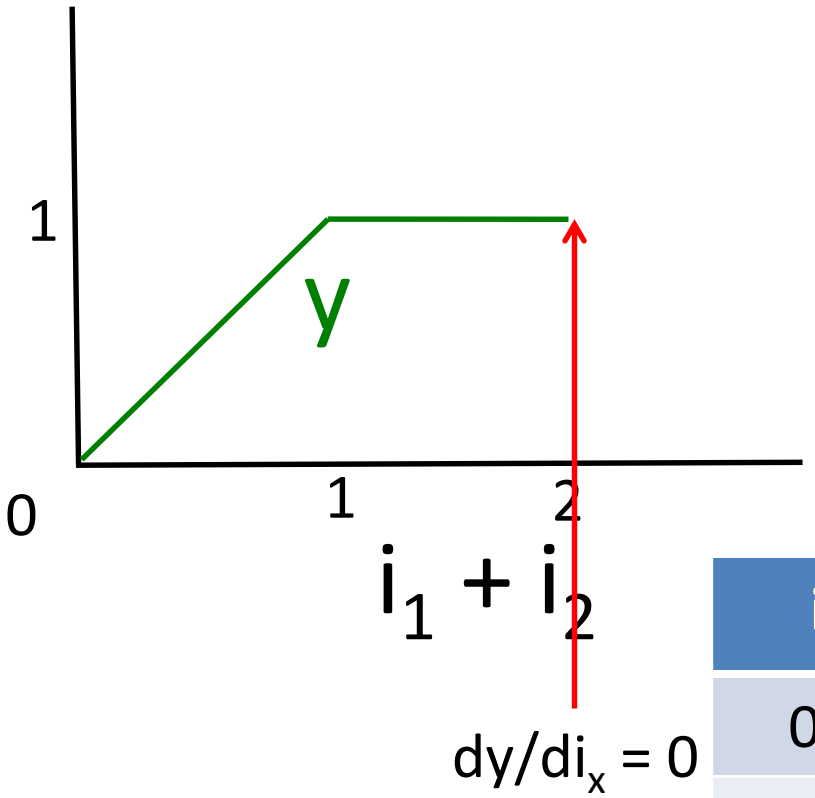


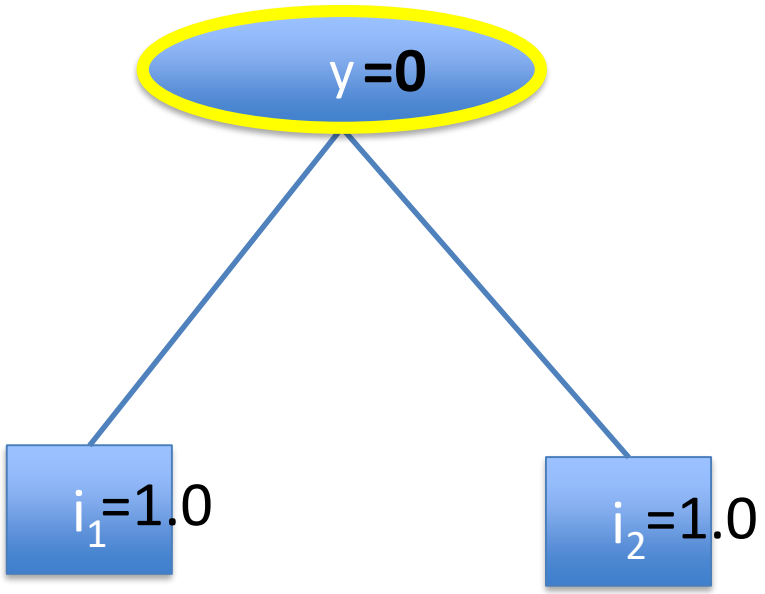


i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1
0.4	0.4	1

i_1	i_2	dy/di_x
0.6	0.6	0
0.8	0.8	0

Integrated Gradients: Another reference-based approach

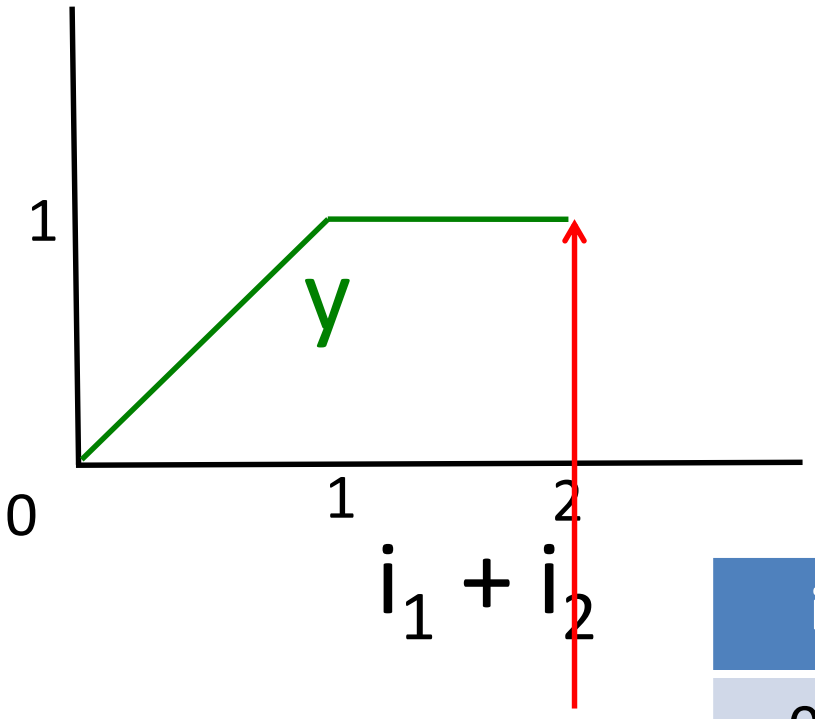




i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1
0.4	0.4	1

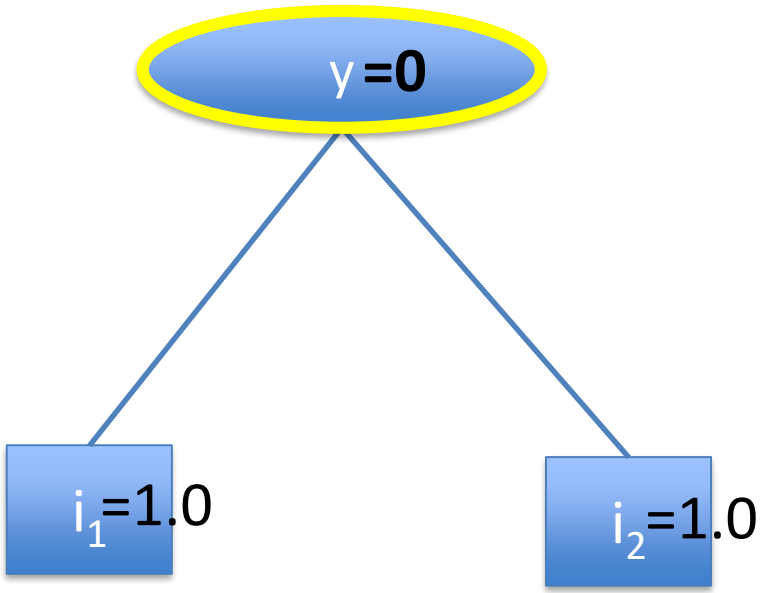
i_1	i_2	dy/di_x
0.6	0.6	0
0.8	0.8	0
1.0	1.0	0

Integrated Gradients: Another reference-based approach



Average $dy/di_x = 0.5$

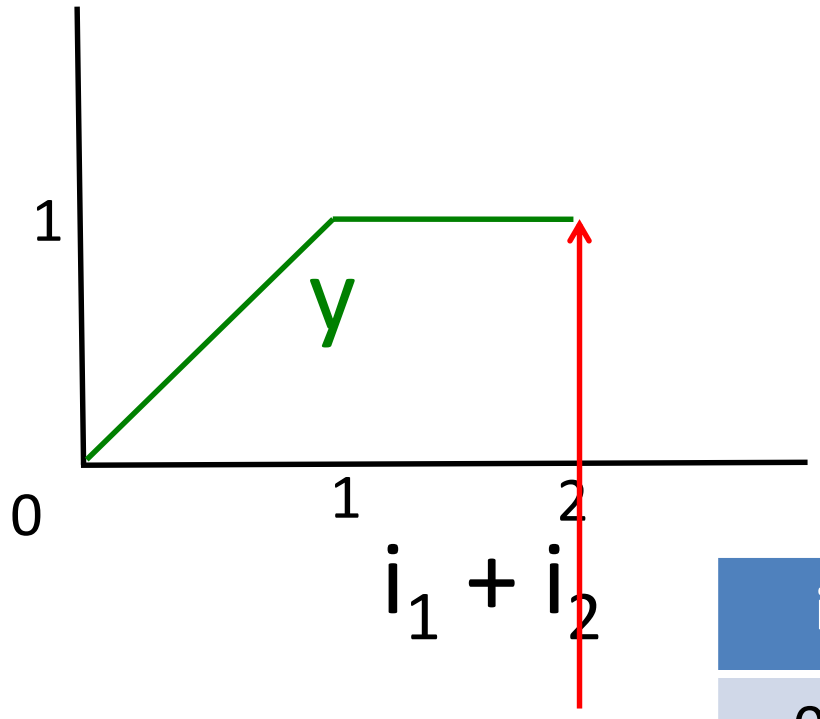
$dy/di_x = 0$

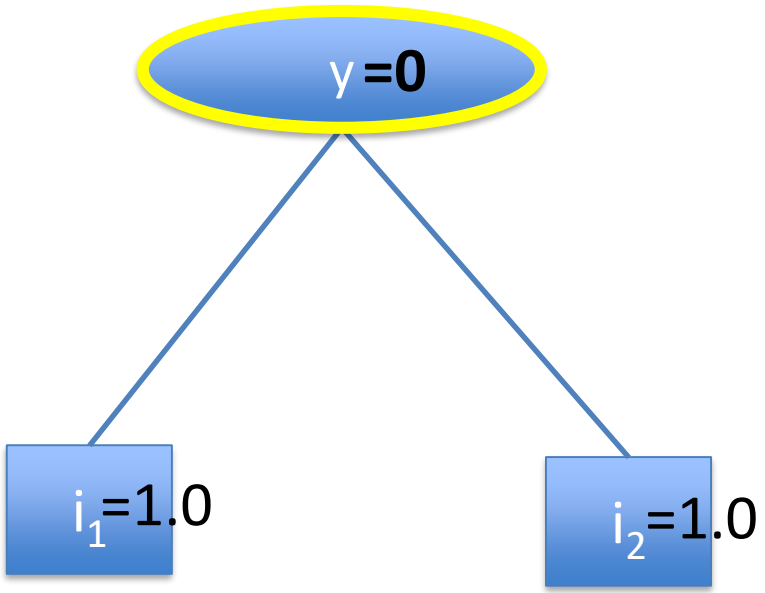


i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1
0.4	0.4	1

i_1	i_2	dy/di_x
0.6	0.6	0
0.8	0.8	0
1.0	1.0	0

Integrated Gradients: Another reference-based approach





i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1
0.4	0.4	1

i_1	i_2	dy/di_x
0.6	0.6	0
0.8	0.8	0
1.0	1.0	0

$dy/di_x = 0$

Average $dy/di_x = 0.5$

$(\text{Average } dy/di_1)*\Delta i_1 = 0.5$

$(\text{Average } dy/di_1)*\Delta i_2 = 0.5$

Integrated Gradients: Another reference-based approach

- Sundararajan et al.

Integrated Gradients: Another reference-based approach

- Sundararajan et al.
- Pros:
  - completely black-box

Integrated Gradients: Another reference-based approach

- Sundararajan et al.
- Pros:
  - completely black-box
  - functionally equivalent networks guaranteed to give the same result

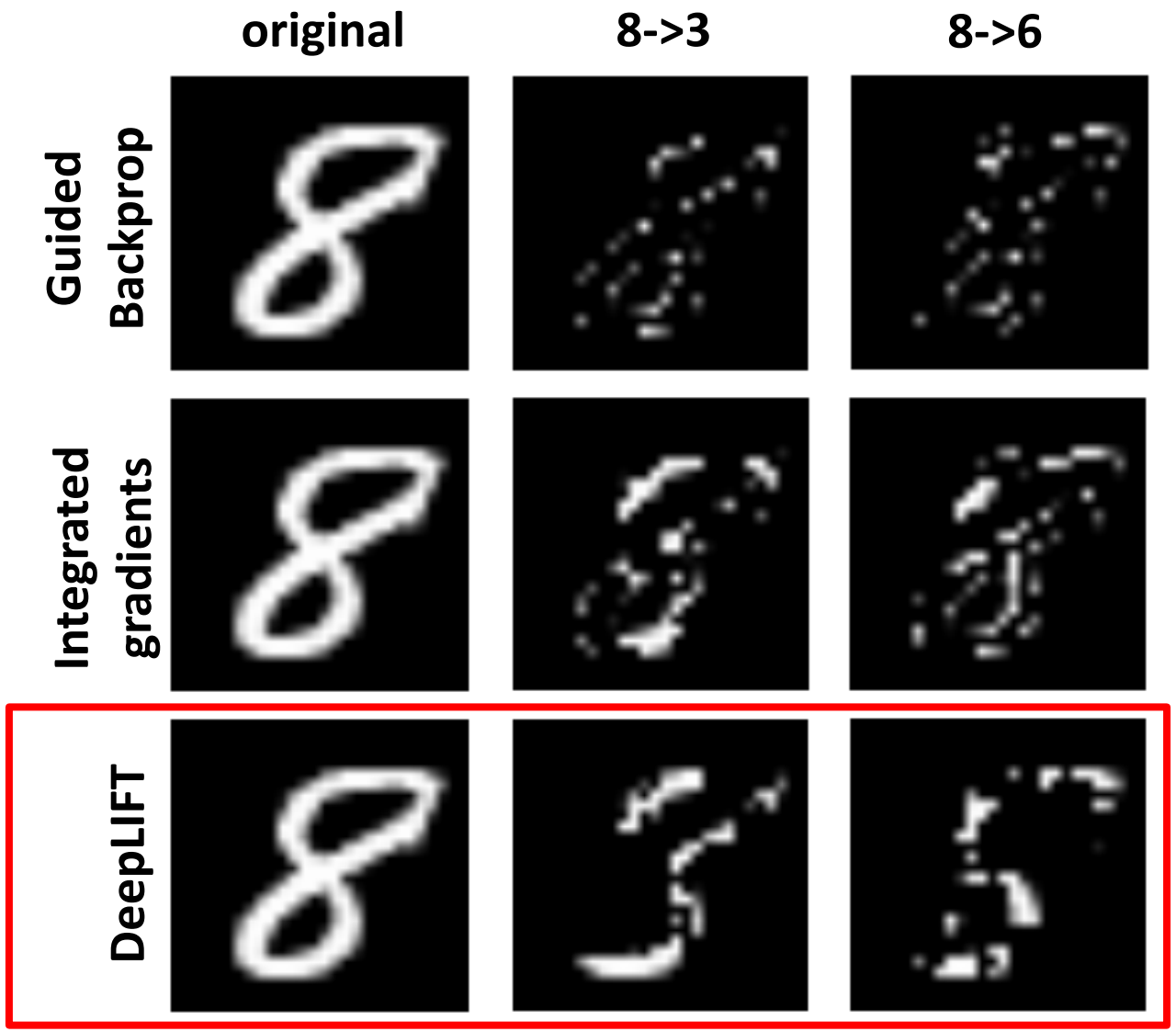
Integrated Gradients: Another reference-based approach

- Sundararajan et al.
- Pros:
  - completely black-box
  - functionally equivalent networks guaranteed to give the same result
- Cons:
  - Repeated gradient calc. adds computational overhead

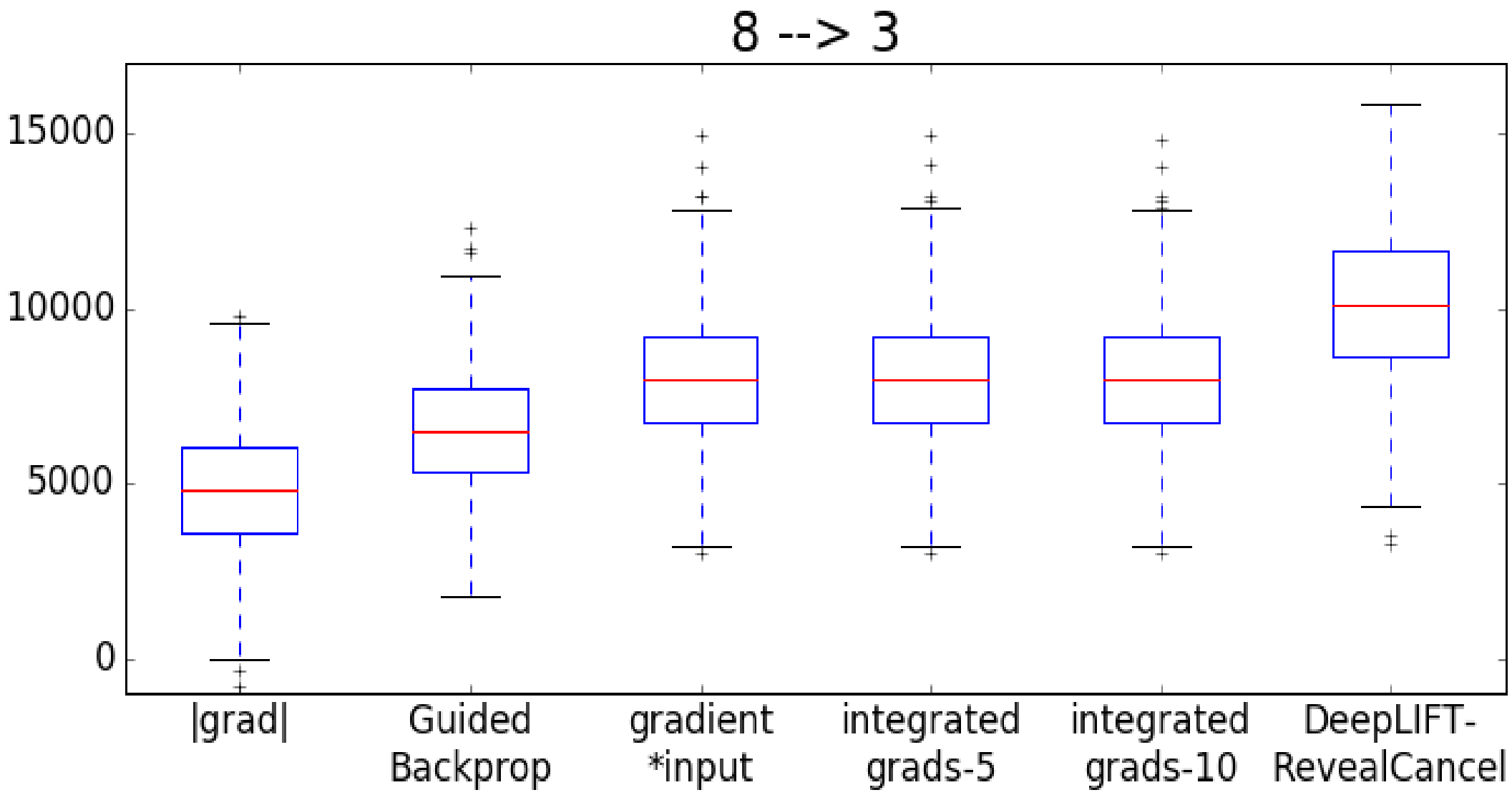
Integrated Gradients: Another reference-based approach

- Sundararajan et al.
- Pros:
  - completely black-box
  - functionally equivalent networks guaranteed to give the same result
- Cons:
  - Repeated gradient calc. adds computational overhead
  - Doesn’t always give an intuitive answer…

morphing 8 to a 3 or a 6



Change in log-odds after morphing



Choice of reference matters!

CIFAR10 model, class = “ship”

Original

Choice of reference matters!

CIFAR10 model, class = “ship”

Original | Reference | DeepLIFT scores

Choice of reference matters!

CIFAR10 model, class = “ship”

Suggestions on how to pick a reference:

- MNIST: all zeros (background)

Choice of reference matters!

CIFAR10 model, class = "ship"

Suggestions on how to pick a reference:

- MNIST: all zeros (background)
- **Consider using a distribution of references**
  - E.g. multiple references generated by shuffling a genomic sequence

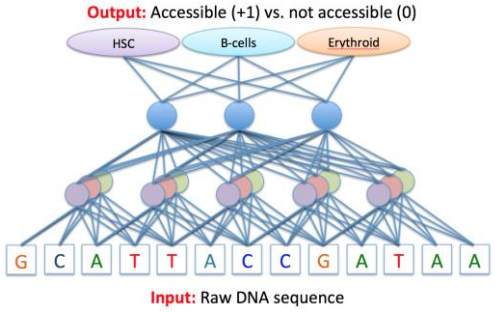
Genomics: DeepLIFT highlights essential DNA bases

Input

Genomics: DeepLIFT highlights essential DNA bases

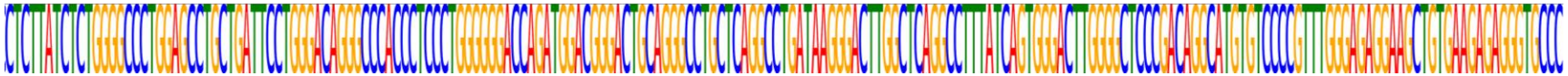
Input

Model + DeepLIFT/gradient metnods

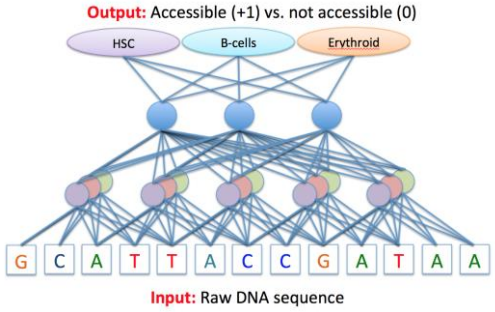


Genomics: DeepLIFT highlights essential DNA bases

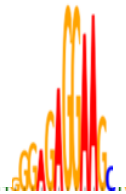
Input



Model + DeepLIFT/gradient metnods



Output



DeepLIFT captures essential features that gradients miss

Simulation: random background sequence with and/or

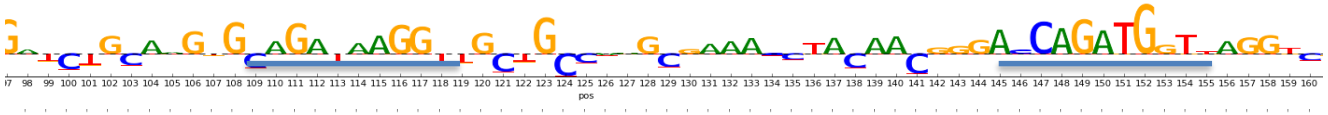
Positive set: BOTH AND

DeepLIFT captures essential features that gradients miss

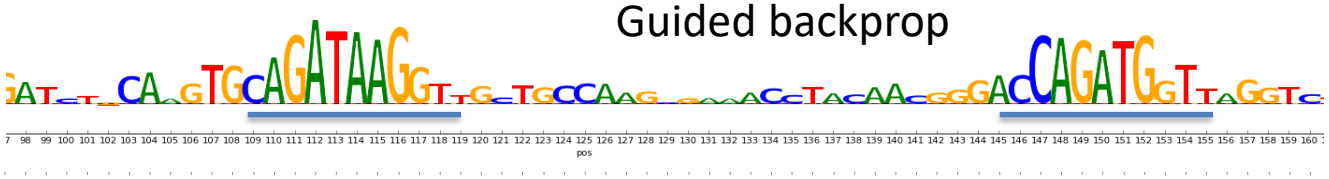
Simulation: random background sequence with and/or

Positive set: BOTH AND

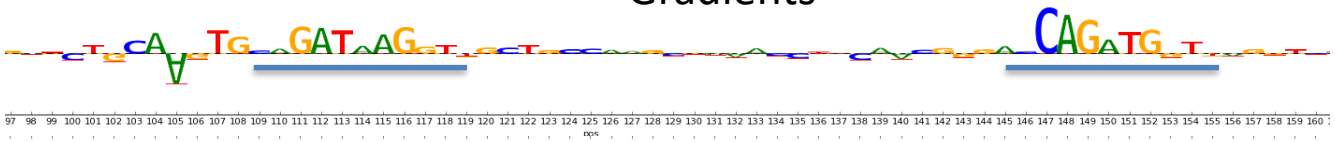
Deconv nets



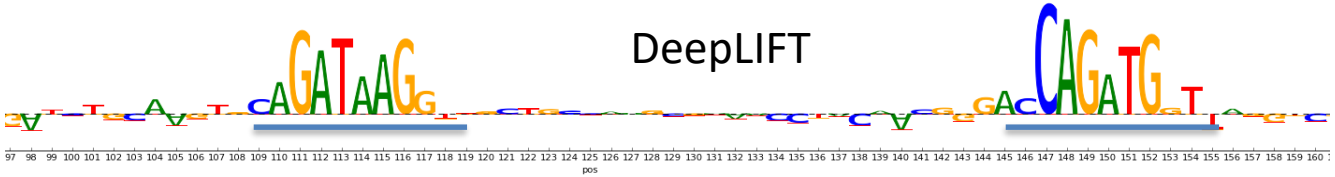
Guided backprop



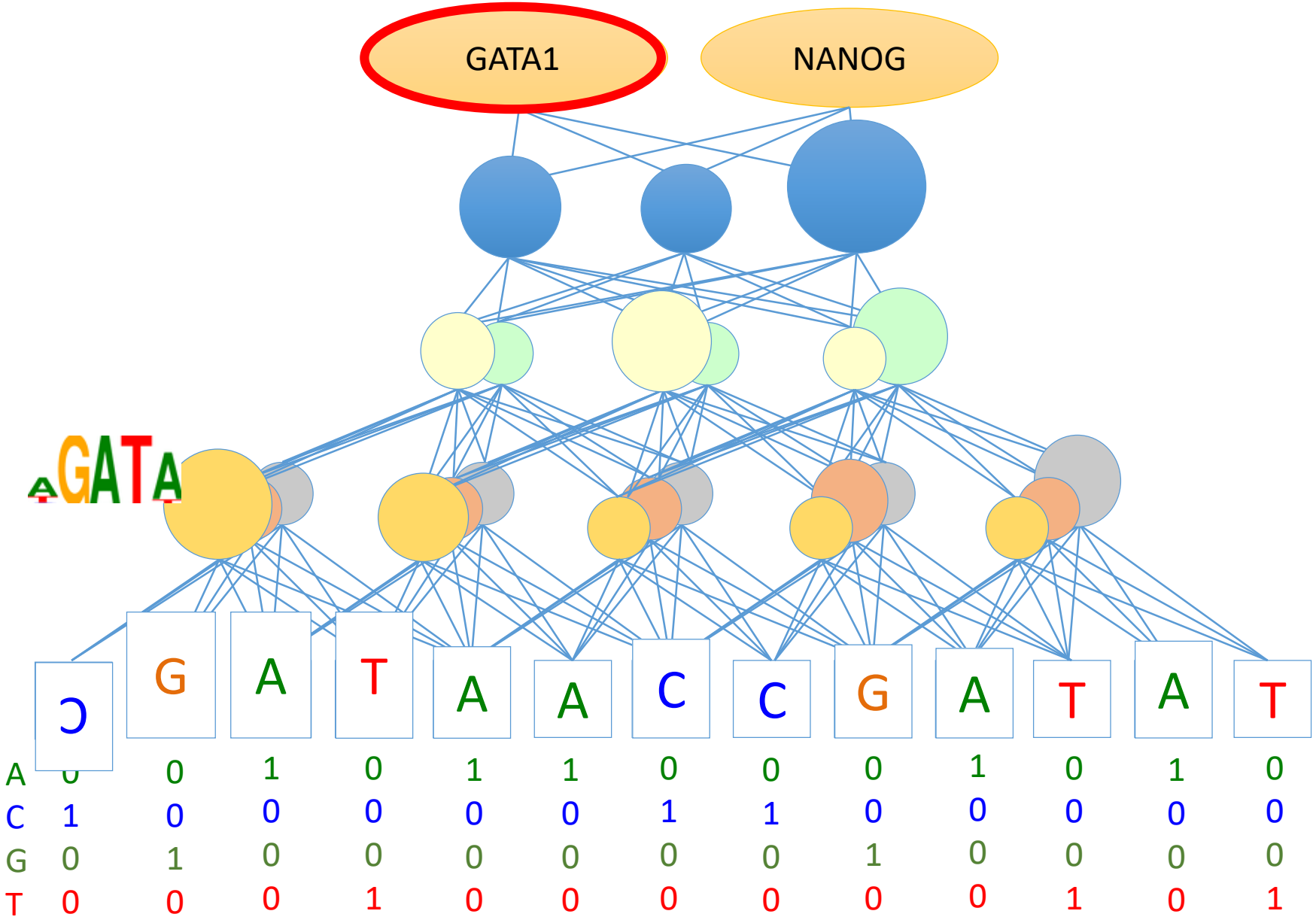
Gradients



DeepLIFT

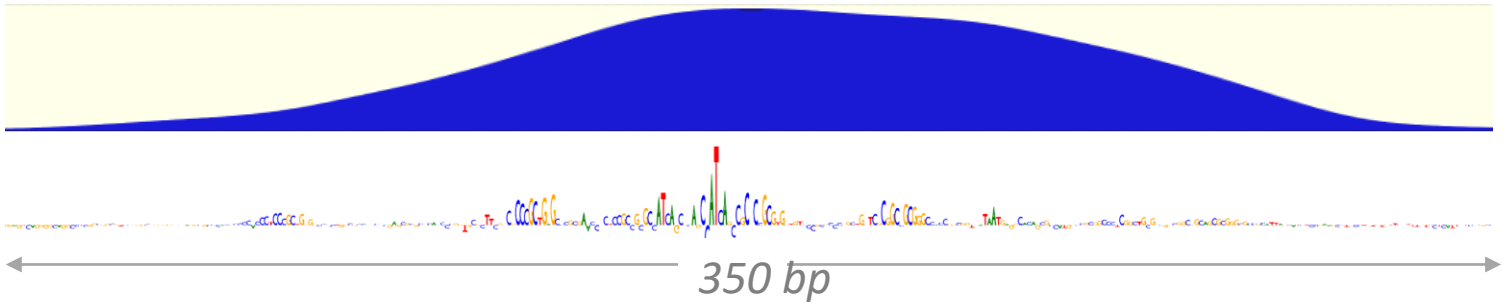


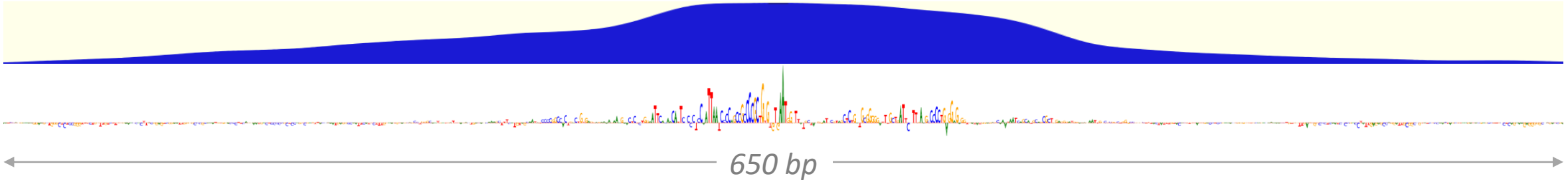
Predicting DNA encoding of TF binding

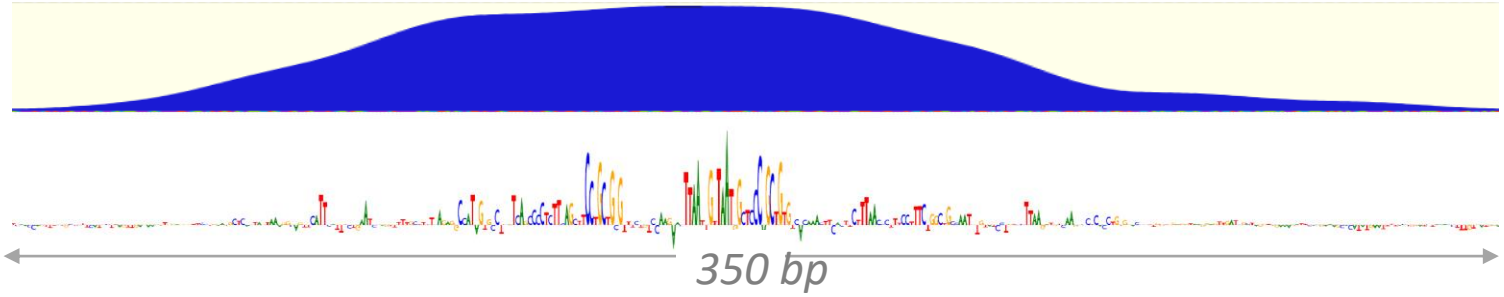


Input: DNA sequence represented as ones and zeros

Inferring high-res binding events and sequence grammars from low-res (300-600 bp) *NANOG* ChIP-seq peaks in hESCs







Nuc. level importance (height of letter) shows coordination of multiple point binding events

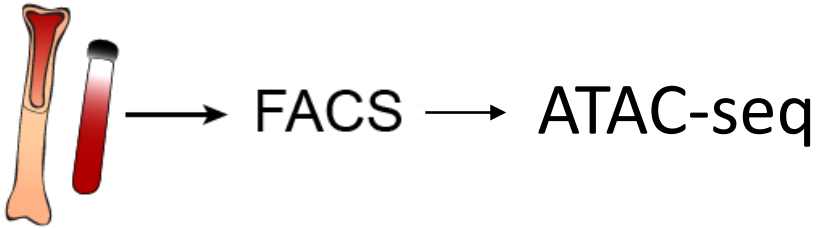
(Currently validating using high-resolution ChIP-exo / ChIP-nexus data)

Dynamic chromatin accessibility in hematopoiesis

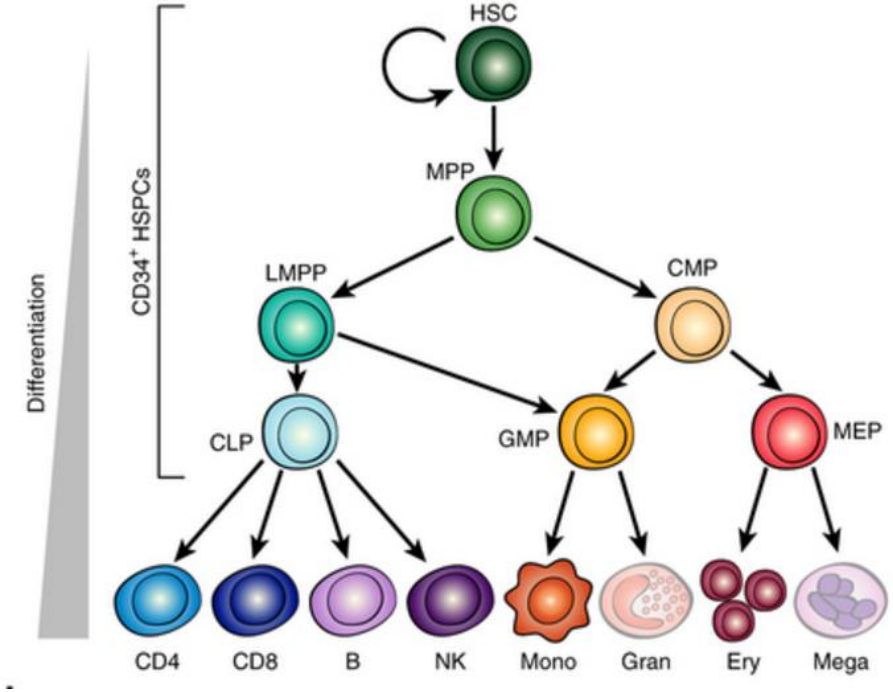
NATURE GENETICS | ARTICLE

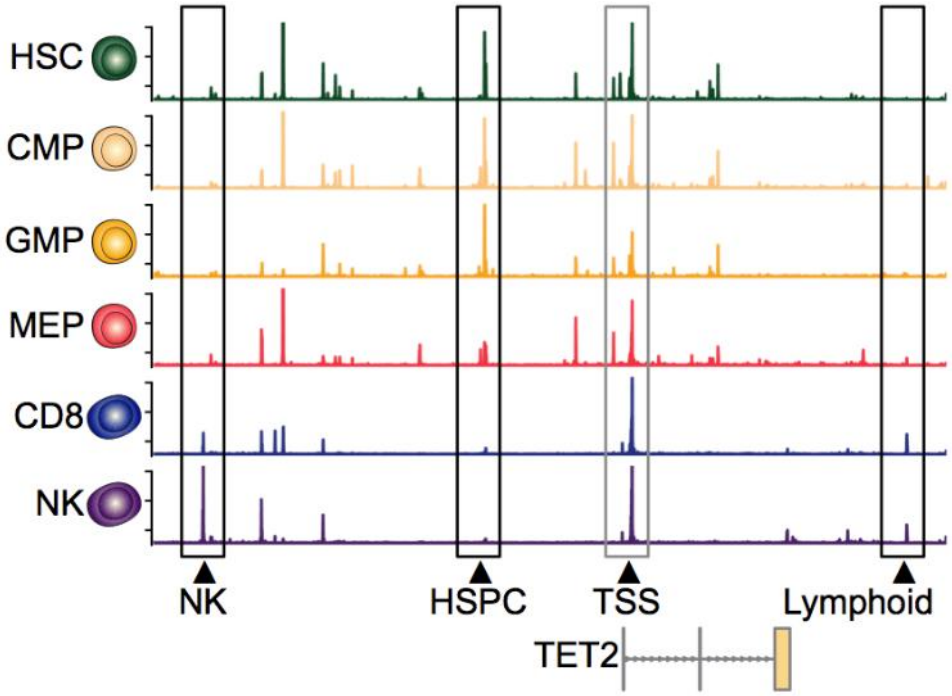
Lineage-specific and single-cell chromatin accessibility charts human hematopoiesis and leukemia evolution

M Ryan Corces, Jason D Buenrostro, Beijing Wu, Peyton G Greenside, Steven M Chan, Julie L Koenig, Michael P Snyder, Jonathan K Pritchard, Anshul Kundaje, William J Greenleaf, Ravindra Majeti & Howard Y Chang

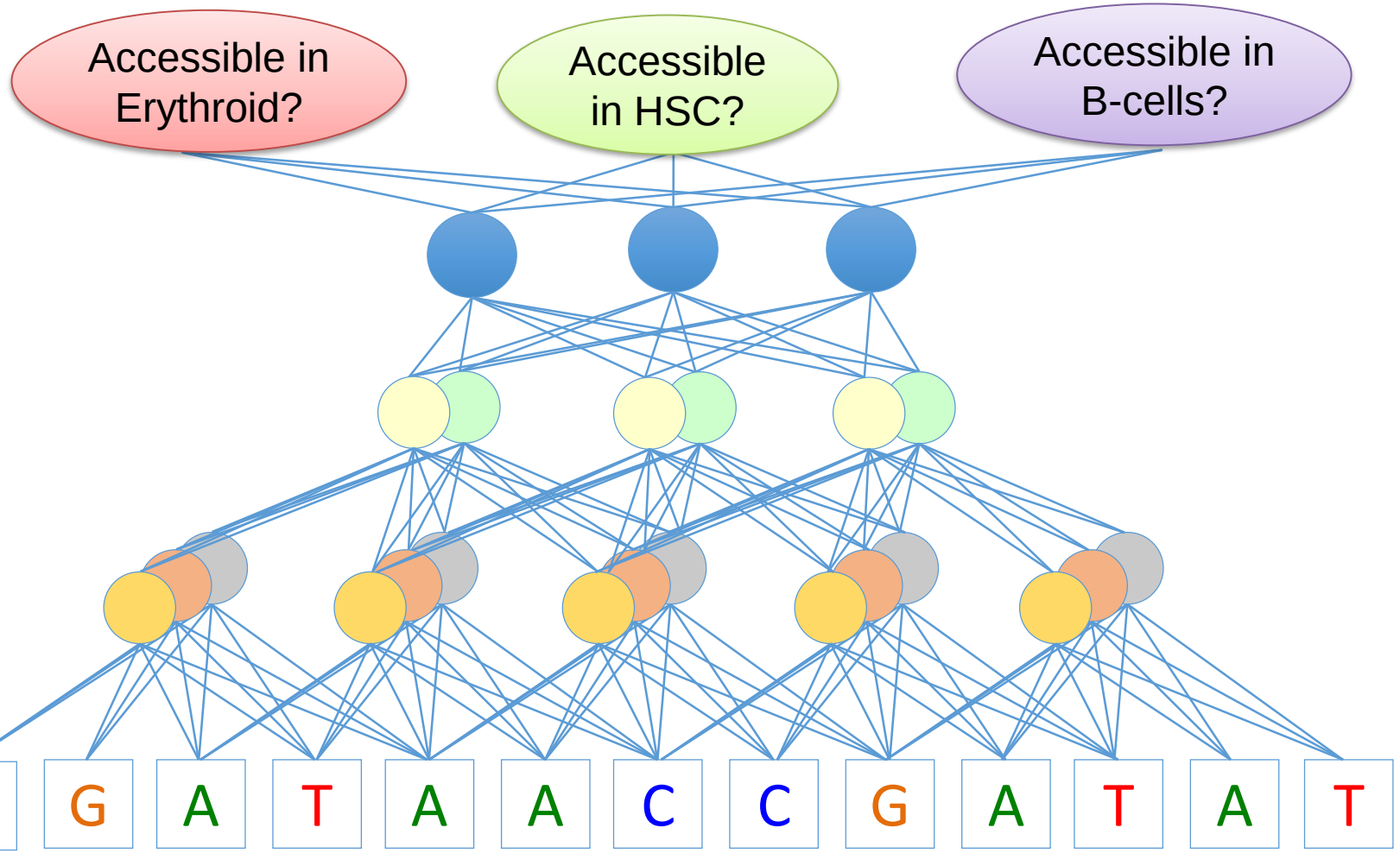


~ 500K ATAC-seq peaks across all cell types





Multi-task deepCNN to jointly predict accessibility across all cell types



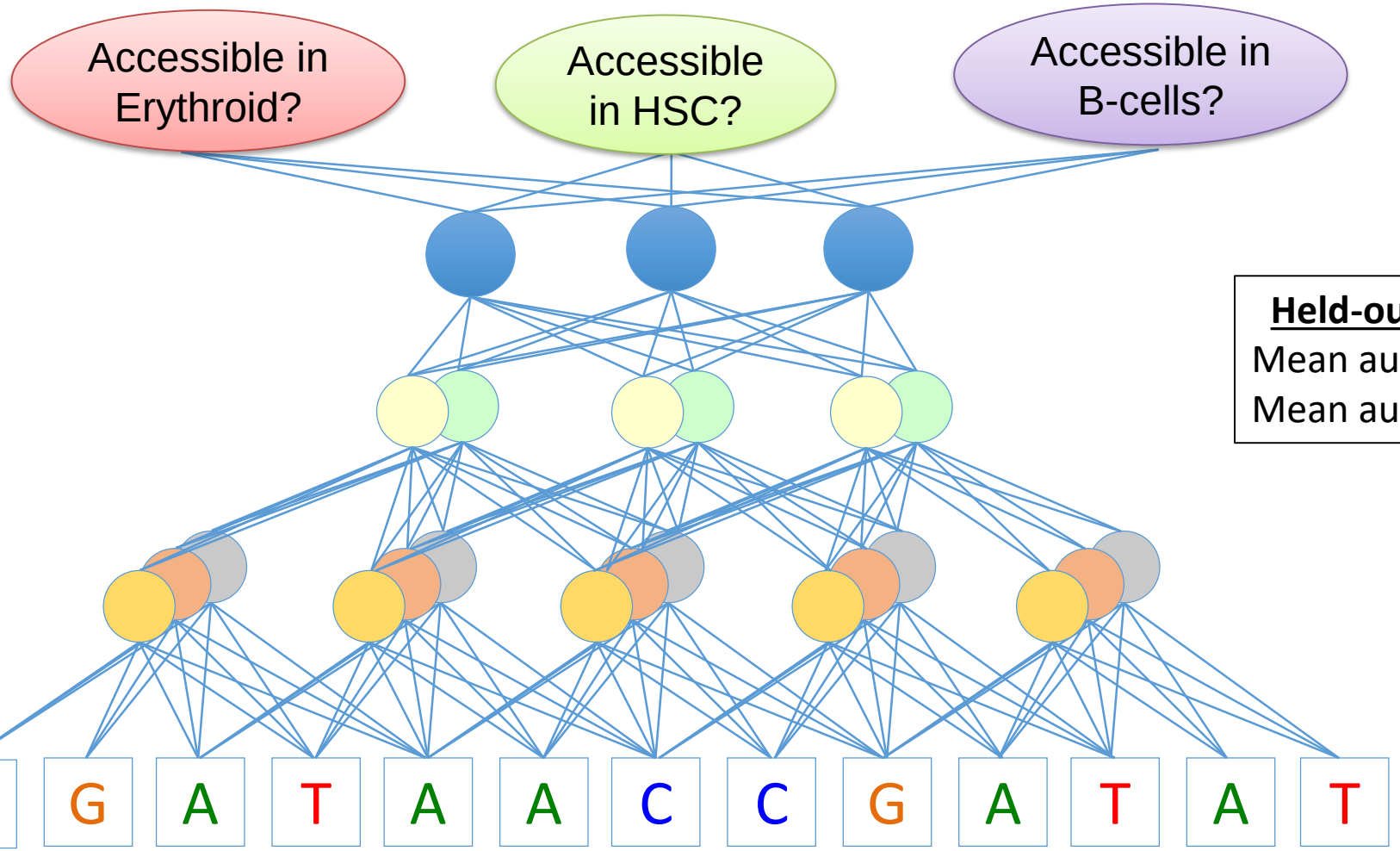
	C	G	A	T	A	A	C	C	G	A	T	A	T
A	0	0	1	0	1	1	0	0	0	1	0	1	0
C	1	0	0	0	0	0	1	1	0	0	0	0	0
G	0	1	0	0	0	0	0	0	1	0	0	0	0
T	0	0	0	1	0	0	0	0	0	0	1	0	1

One-hot encoded input: DNA sequence represented as ones and zeros

Peyton Greenside

Similar to
Kelley et al. 2016 (Basset)
Zhou et al. 2015 (DeepSEA)

Multi-task deepCNN to jointly predict accessibility across all cell types



	C	G	A	T	A	A	C	C	G	A	T	A	T
A	0	0	1	0	1	1	0	0	0	1	0	1	0
C	1	0	0	0	0	0	1	1	0	0	0	0	0
G	0	1	0	0	0	0	0	0	1	0	0	0	0
T	0	0	0	1	0	0	0	0	0	0	1	0	1

One-hot encoded input: DNA sequence represented as ones and zeros

Held-out peaks prediction accuracy

Mean auROC = 0.82

Mean auPRC = 0.65

Peyton Greenside

Similar to
Kelley et al. 2016 (Basset)
Zhou et al. 2015 (DeepSEA)

Distinct TF grammars can activate same control element in different tissues

Regulatory element

Peyton Greenside

Distinct TF grammars can activate same control element in different tissues

Regulatory element

DeepLIFT scores
Tissue: **Hema stem cells**

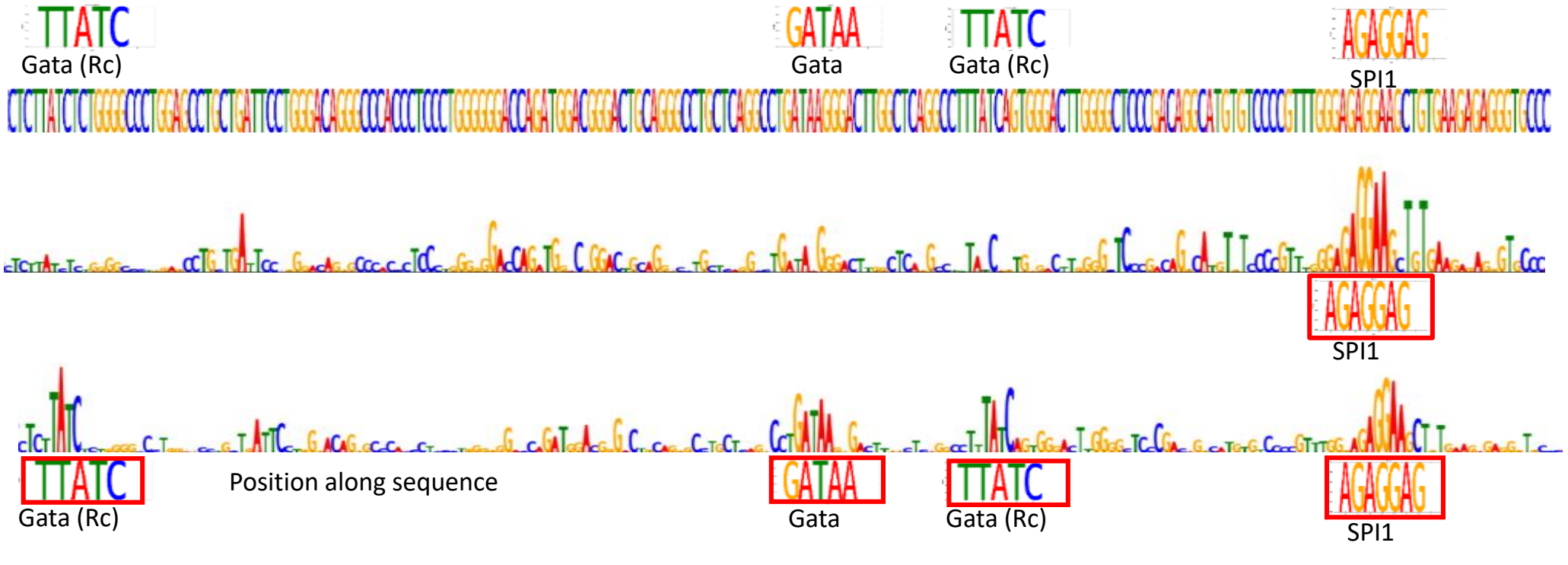
Peyton Greenside

Distinct TF grammars can activate same control element in different tissues

Regulatory element

DeepLIFT scores
Tissue: **Hema stem cells**

DeepLIFT scores
Tissue: **Erythroid cells**



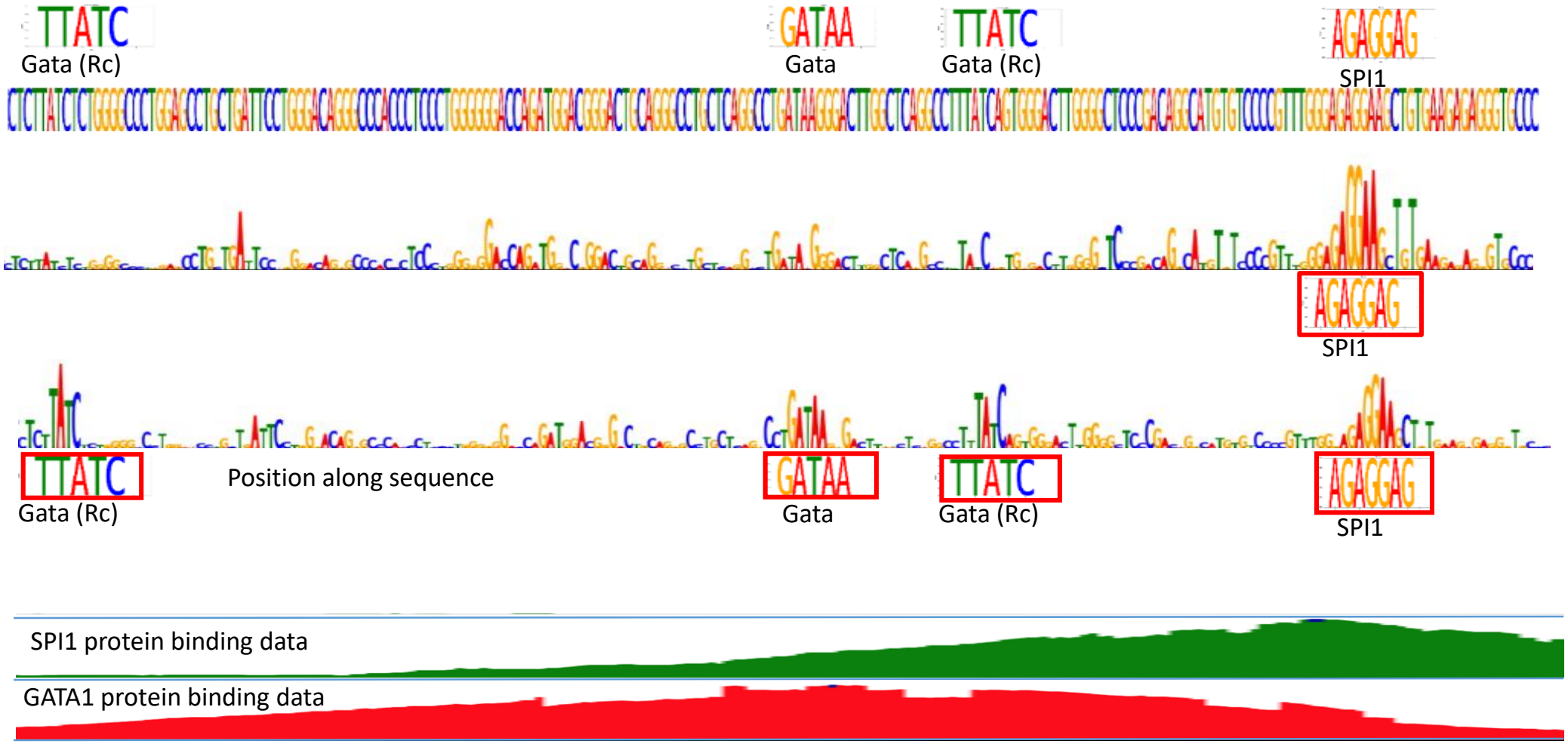
Peyton Greenside

Distinct TF grammars can activate same control element in different tissues

Regulatory element

DeepLIFT scores
Tissue: **Hema stem cells**

DeepLIFT scores
Tissue: **Erythroid cells**

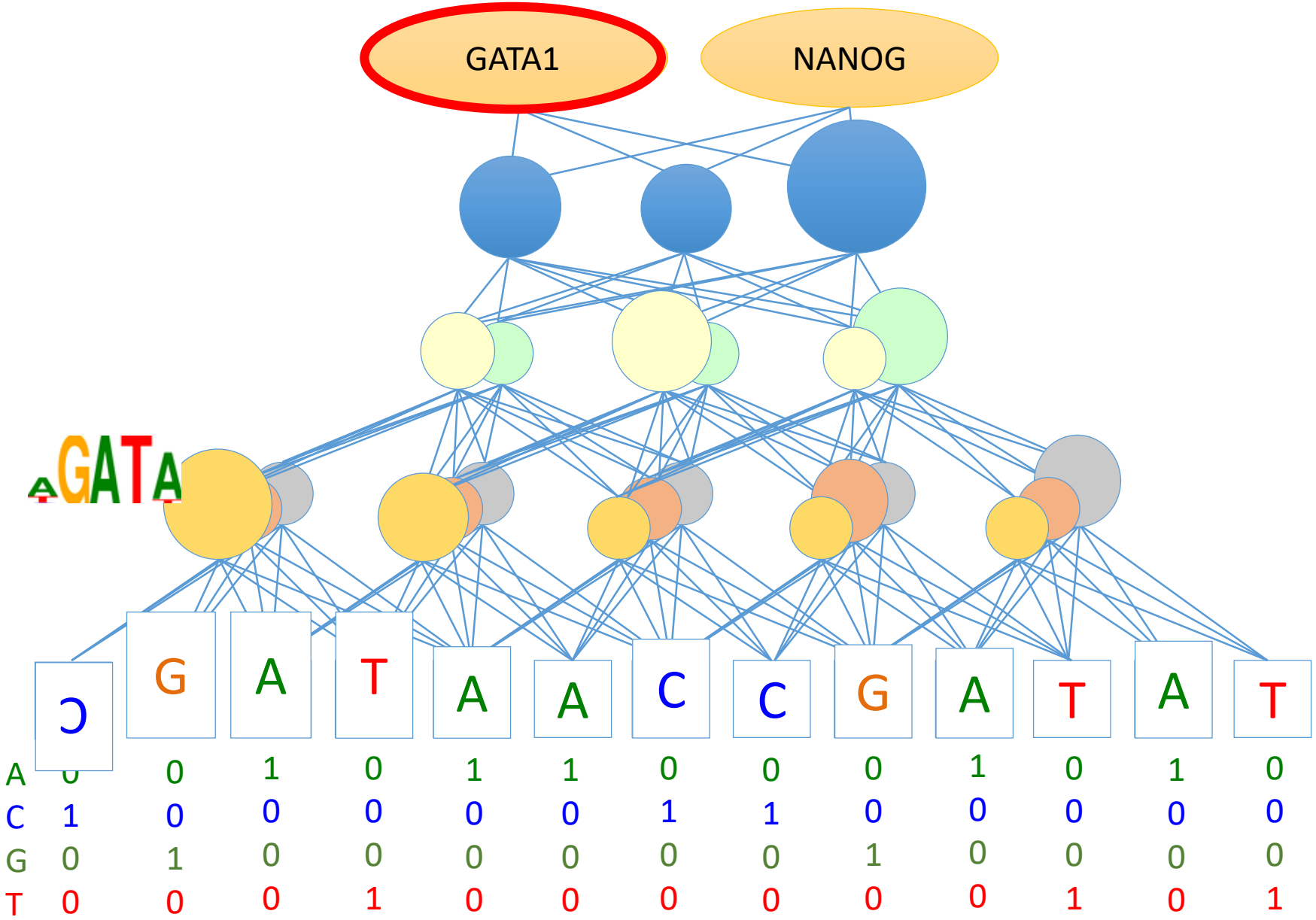


Validation: ChIP-seq in Erythroid cells

Peyton Greenside

Global pattern discovery

Can we interpret conv. filters as motif patterns?



Input: DNA sequence represented as ones and zeros

Identifying important conv. filters

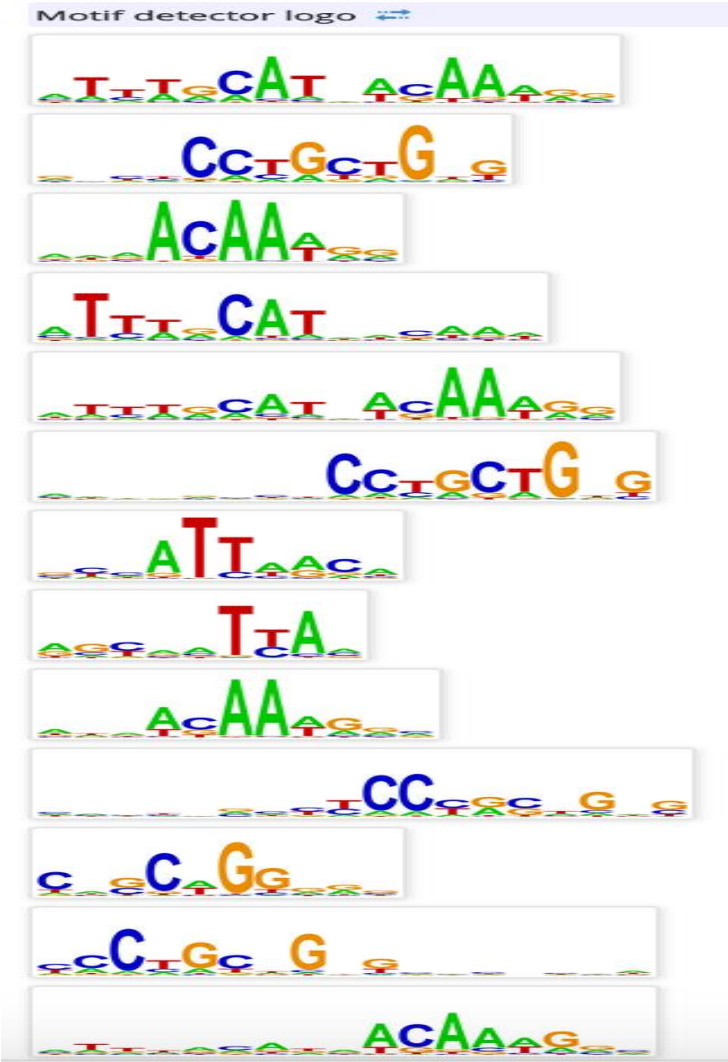
- Pattern-detector knockout
  - Convolutional layer has pattern detectors, called “filters”
  - Basset:
    - “We nullified each filter by setting its output to the mean value, which obstructs it from passing any information forward through the network. We computed the L2 norm of the change in predicted accessibility to measure its influence.”
    - Conceptually similar to “mutagenesis on the filters”
    - Computationally inefficient!
    - From the Basset tutorial:

> This analysis requires considerably more compute time, so I separated it into a different script. To give it a sufficient number of sequences to obtain a good estimate influence, I typically run it overnight. If your computer is using too much memory, decrease the batch size. I'm going to run here with 1,000 sequences, but I used 20,000 for the paper.

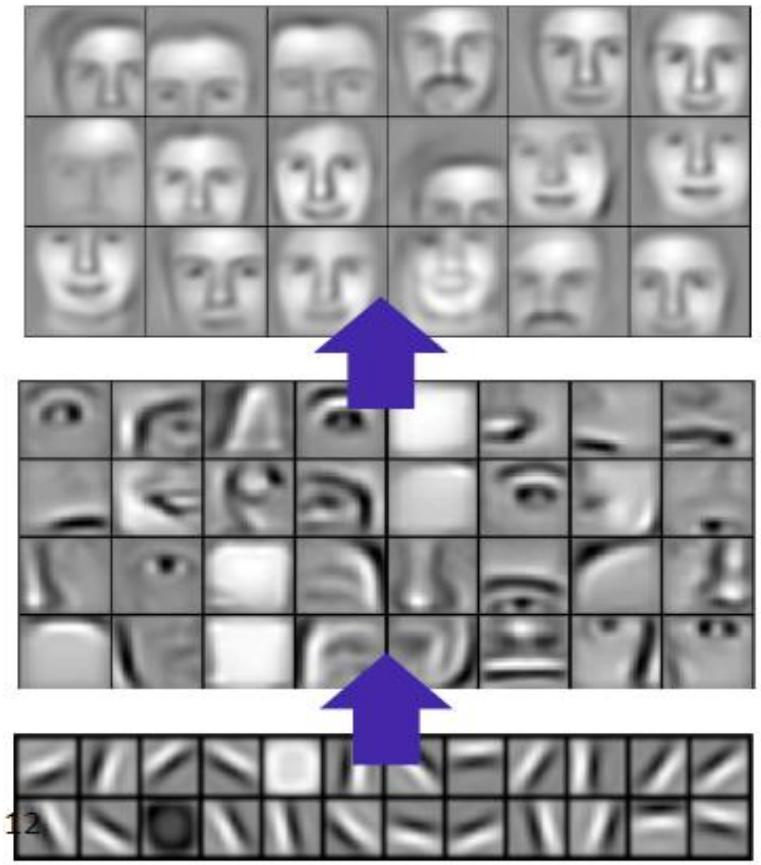
    - Buffering still a problem

Visualizing individual convolutional neurons gives highly distributed representations

DeepBind (Alipanahi et al.) for Nanog



In computer vision

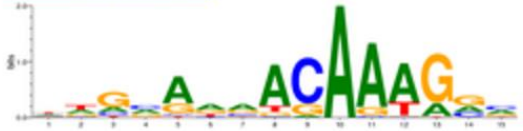


Known motif databases also have this redundancy issue

ENCODE PWMs
(Kheradpour et al.)

NANOG_disc1:

NANOG_disc2:



NANOG_disc3:

NANOG_disc4:

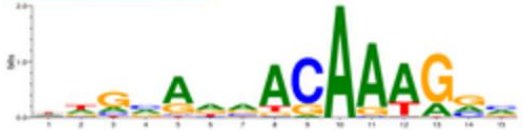
NANOG_known1:

Known motif databases also have this redundancy issue

ENCODE PWMs
(Kheradpour et al.)

NANOG_disc1:

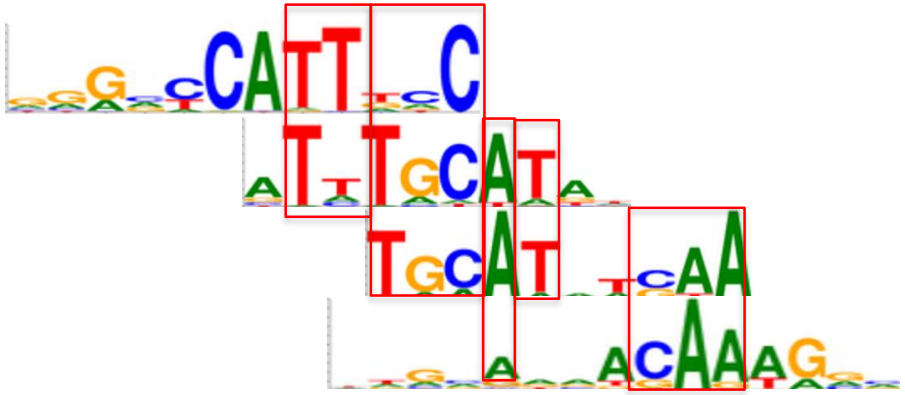
NANOG_disc2:



NANOG_disc3:

NANOG_disc4:

NANOG_known1:

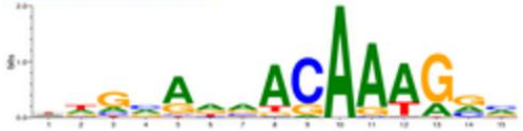


Known motif databases also have this redundancy issue

ENCODE PWMs
(Kheradpour et al.)

NANOG_disc1:

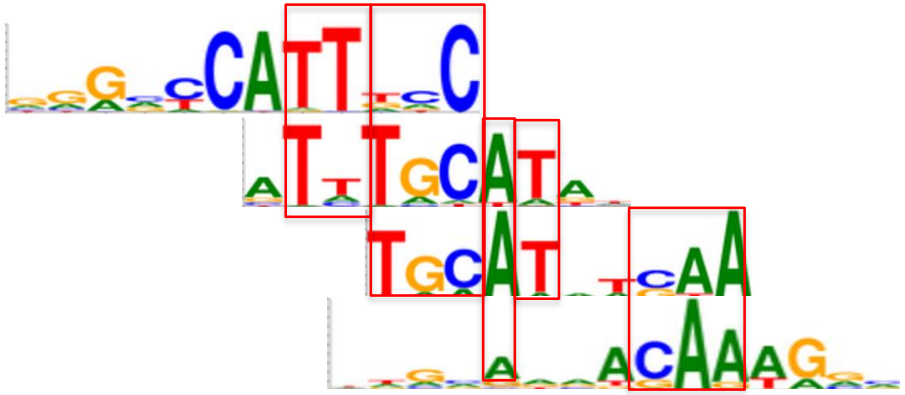
NANOG_disc2:



NANOG_disc3:

NANOG_disc4:

NANOG_known1:

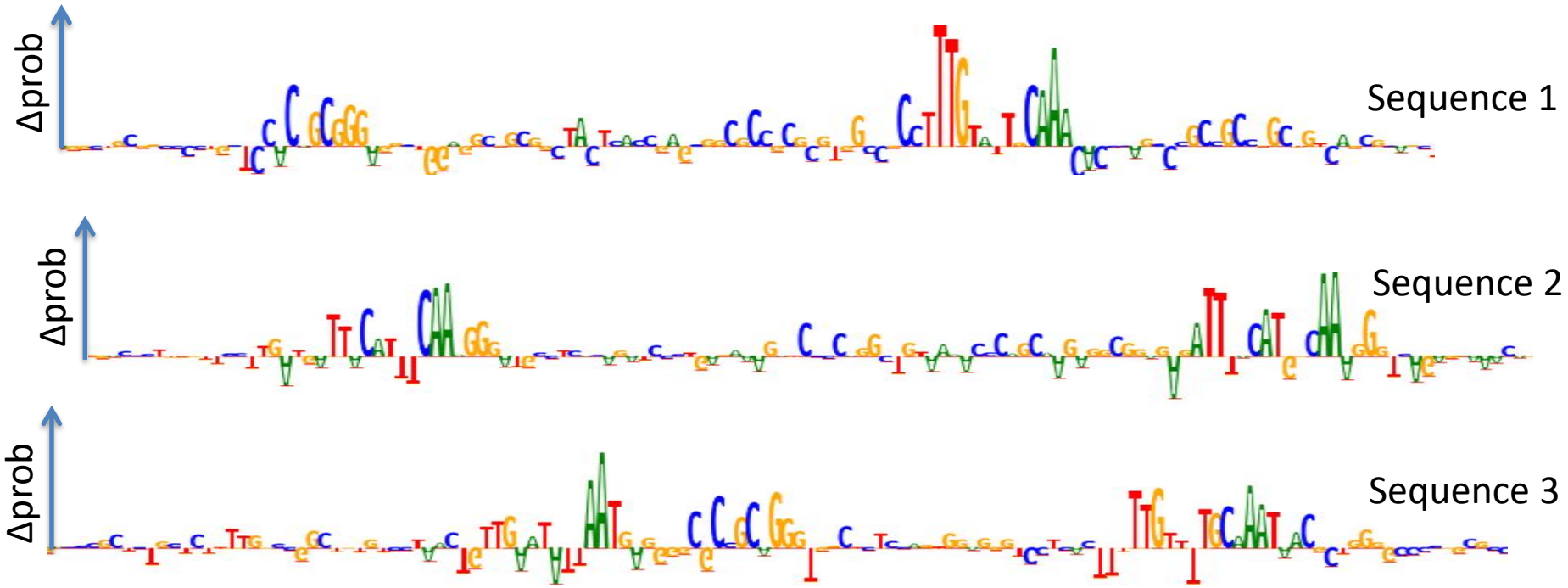


Aggregated pattern

How do we combine the contributions of multiple pattern detectors to find consolidated patterns?

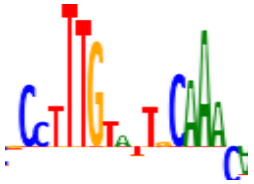
How do we combine the contributions of multiple pattern detectors to find consolidated patterns?

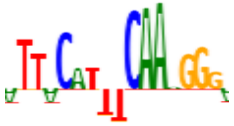
Insight: input-level importance scores reveal combined contributions



How do we combine the contributions of multiple pattern detectors to find consolidated patterns?

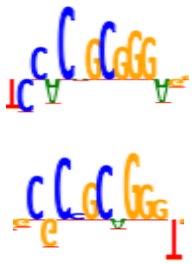
Insight: input-level importance scores reveal combined contributions

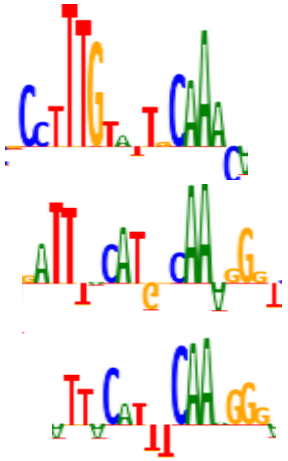




How do we combine the contributions of multiple pattern detectors to find consolidated patterns?

Insight: input-level importance scores reveal combined contributions





How do we combine the contributions of multiple pattern detectors to find consolidated patterns?

Insight: input-level importance scores reveal combined contributions

How do we combine the contributions of multiple pattern detectors to find consolidated patterns?

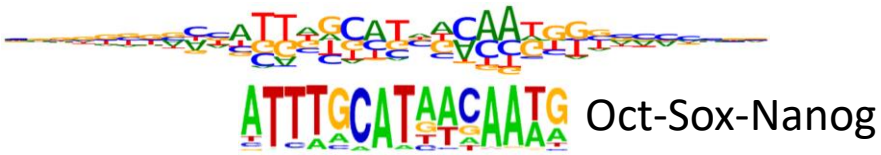
Insight: input-level importance scores reveal combined contributions

MoDISco: Motif Discovery from Importance Scores

MoDISco uncovers broader patterns than existing approaches for the Nanog TF

MoDISco

Known pattern
(Heinz et al.)



Oct-Sox-Nanog

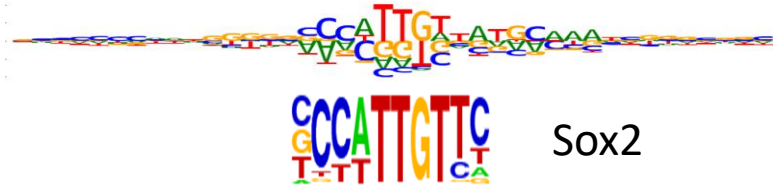
MoDISco

Known pattern
(Heinz et al.)

Nanog

MoDISco

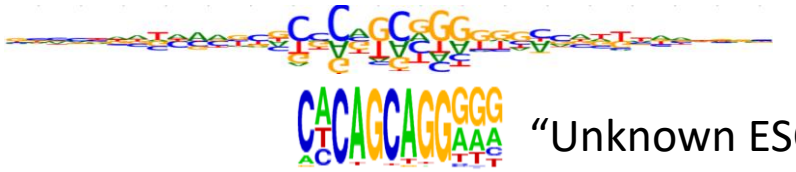
Known pattern
(Heinz et al.)



Sox2

MoDISco

Known pattern
(Heinz et al.)



"Unknown ESC element"

MoDISco uncovers broader patterns than existing approaches for the Nanog TF

MoDISco

Known pattern
(Heinz et al.)

Oct-Sox-Nanog

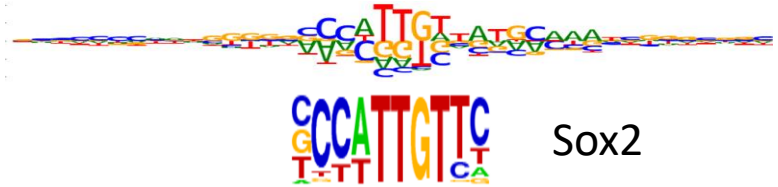
MoDISco

Known pattern
(Heinz et al.)

Nanog

MoDISco

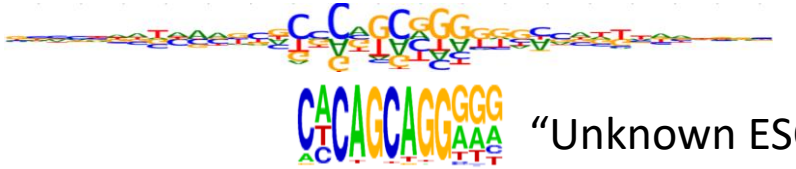
Known pattern
(Heinz et al.)



Sox2

MoDISco

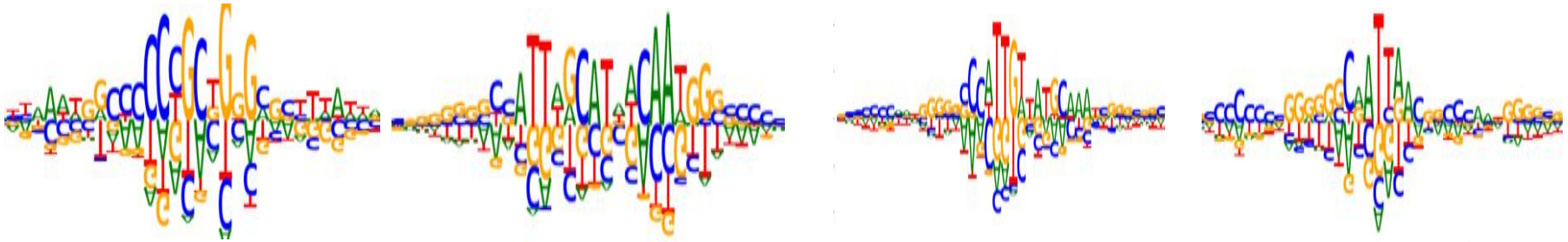
Known pattern
(Heinz et al.)



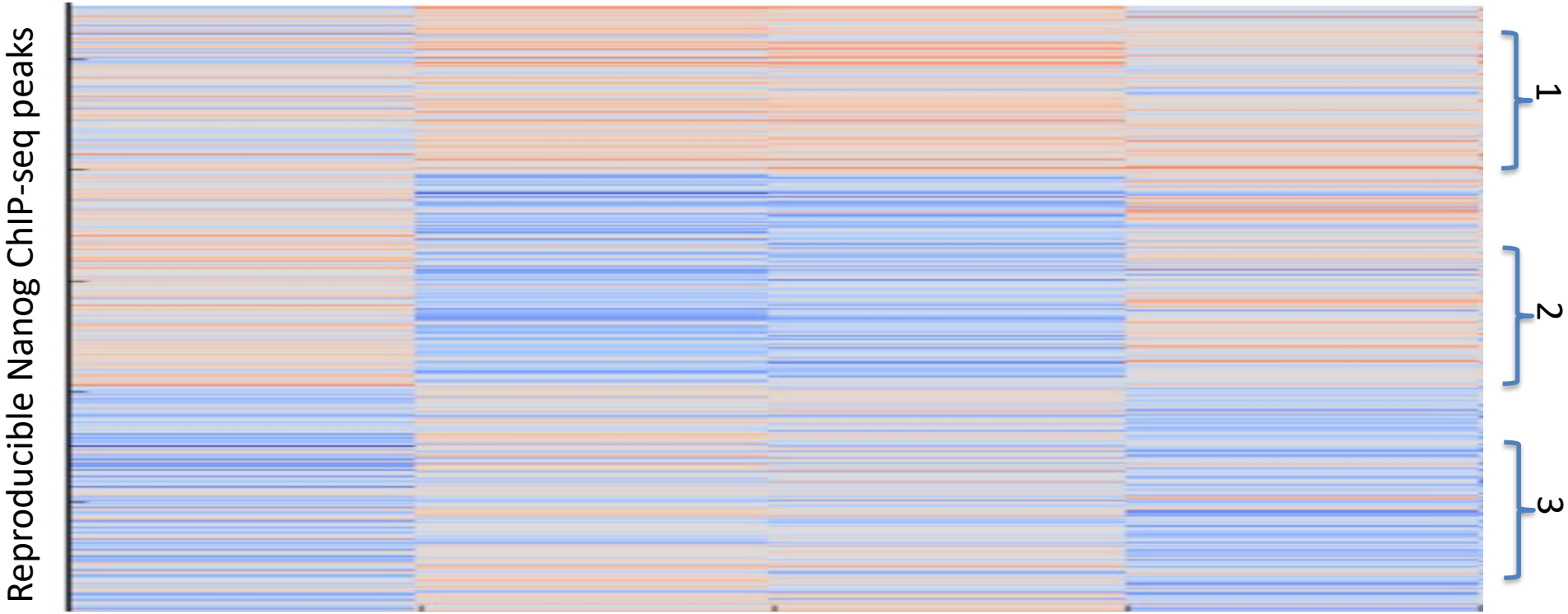
Zic3 (Jaspar; Zhao et al.)

"Unknown ESC element"

At least 3 distinct classes of Nanog sites



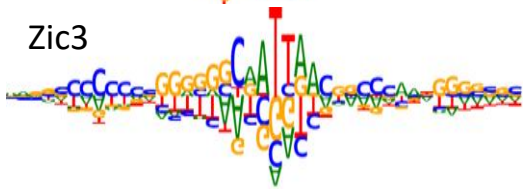
Zic3 motif | Oct4-Sox2-Nanog fusion motif | Sox2 motif | Nanog motif



Potential novel relationship between Nanog and Zic3 discovered

Nanog

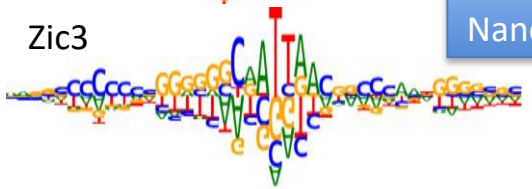
Zic3



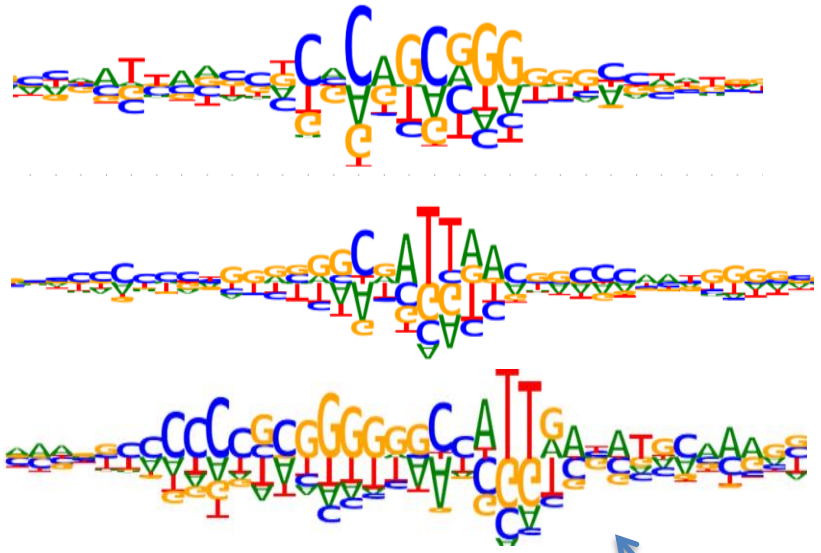
Potential novel relationship between Nanog and Zic3 discovered

Nanog

Zic3



Re-cluster with Nanog/Zic3 patterns only

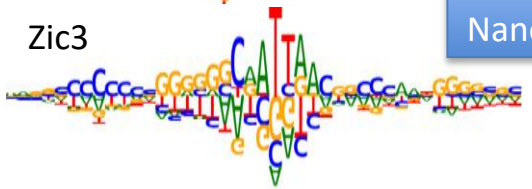


Novel combination of Nanog and Zic3 patterns

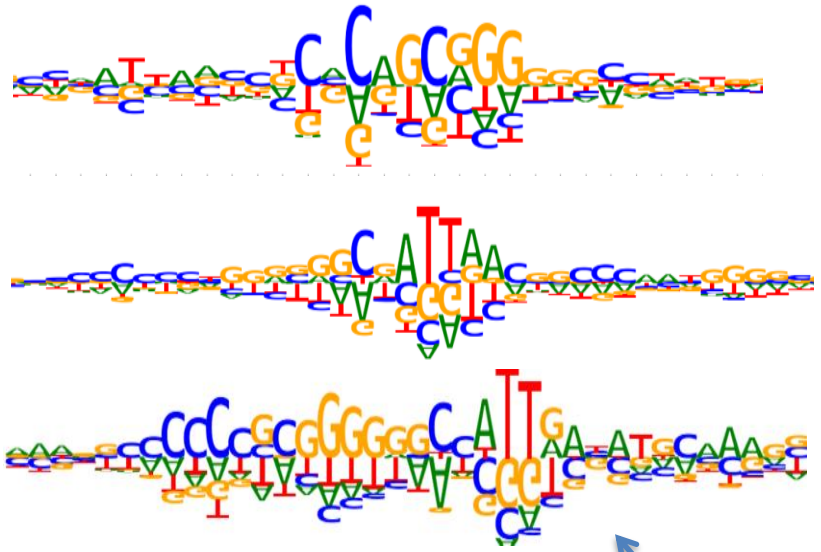
Potential novel relationship between Nanog and Zic3 discovered

Nanog

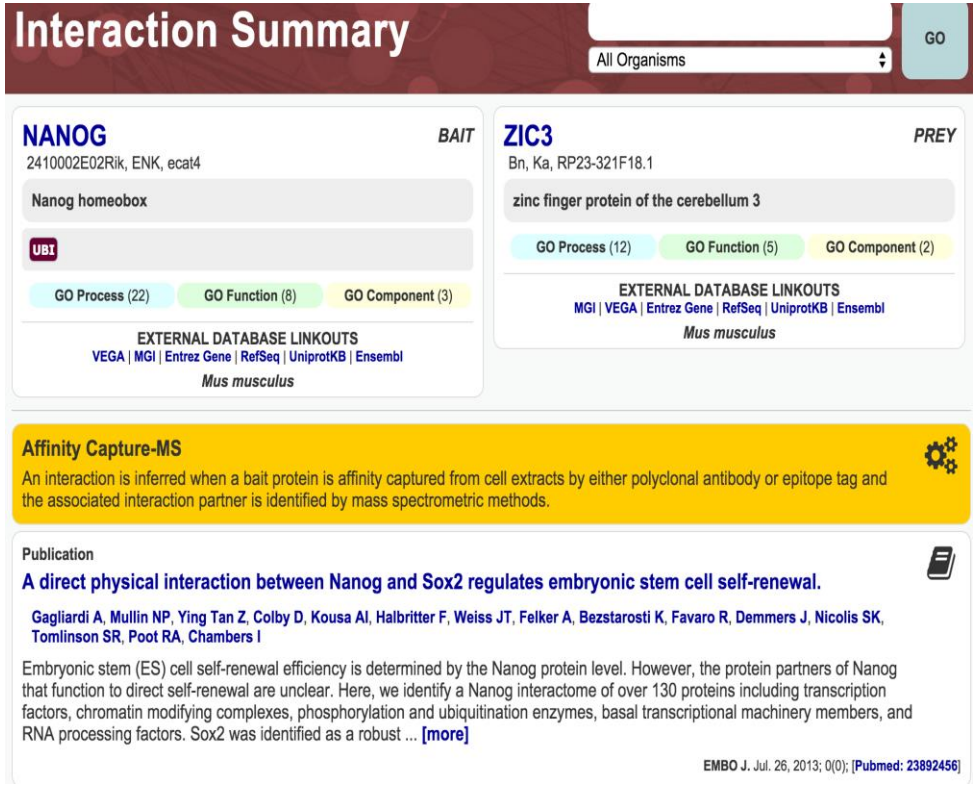
Zic3



Re-cluster with Nanog/Zic3 patterns only



Novel combination of Nanog and Zic3 patterns



Interaction Summary

All Organisms

GO

NANOG BAIT

2410002E02Rik, ENK, ecat4

Nanog homeobox

UBI

GO Process (22) | GO Function (8) | GO Component (3)

EXTERNAL DATABASE LINKOUTS

VEGA | MGI | Entrez Gene | RefSeq | UniprotKB | Ensembl

Mus musculus

ZIC3 PREY

Bn, Ka, RP23-321F18.1

zinc finger protein of the cerebellum 3

GO Process (12) | GO Function (5) | GO Component (2)

EXTERNAL DATABASE LINKOUTS

MGI | VEGA | Entrez Gene | RefSeq | UniprotKB | Ensembl

Mus musculus

Affinity Capture-MS

An interaction is inferred when a bait protein is affinity captured from cell extracts by either polyclonal antibody or epitope tag and the associated interaction partner is identified by mass spectrometric methods.

Publication

A direct physical interaction between Nanog and Sox2 regulates embryonic stem cell self-renewal.

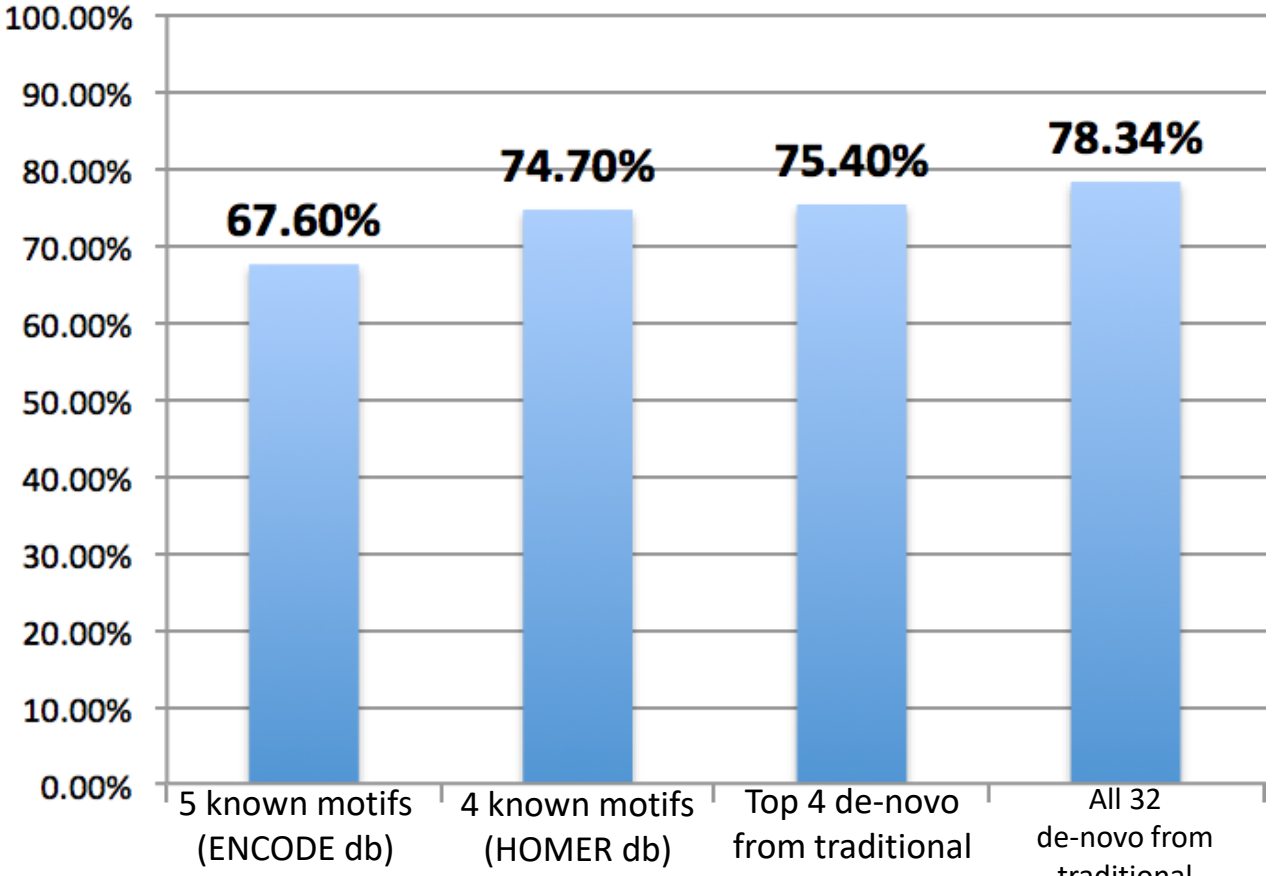
Gagliardi A, Mullin NP, Ying Tan Z, Colby D, Kousa AI, Halbritter F, Weiss JT, Felker A, Bezstarosti K, Favaro R, Demmers J, Nicolis SK, Tomlinson SR, Poot RA, Chambers I

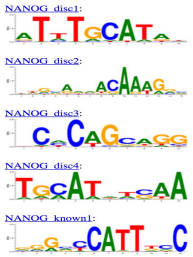
Embryonic stem (ES) cell self-renewal efficiency is determined by the Nanog protein level. However, the protein partners of Nanog that function to direct self-renewal are unclear. Here, we identify a Nanog interactome of over 130 proteins including transcription factors, chromatin modifying complexes, phosphorylation and ubiquitination enzymes, basal transcriptional machinery members, and RNA processing factors. Sox2 was identified as a robust ... [more]

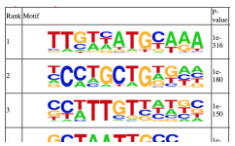
EMBO J. Jul. 26, 2013; 0(0); [Pubmed: 23892456]

MoDISco patterns outperform traditional motif discovery methods

Logistic regression on top hits to each pattern, auROC

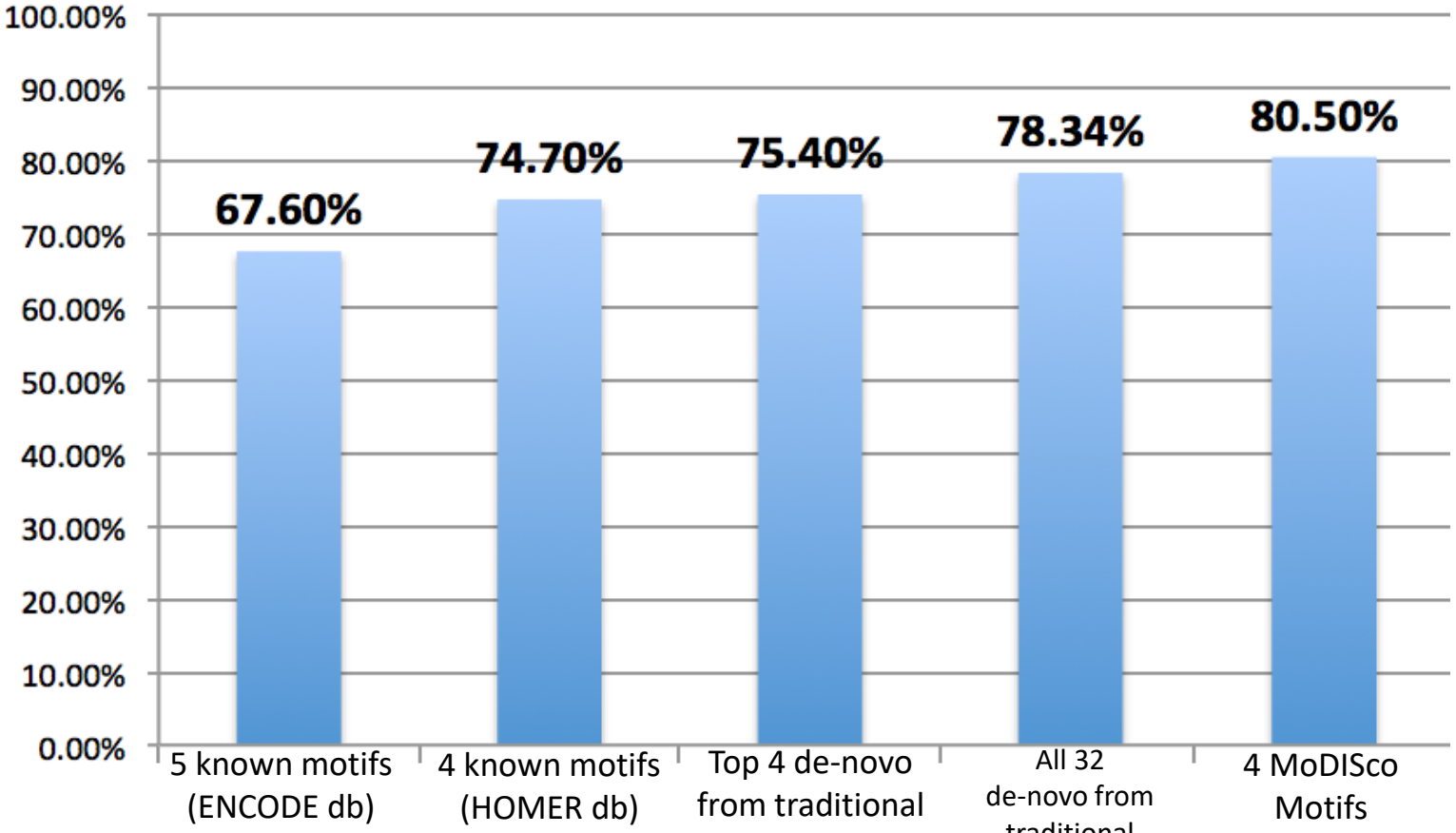


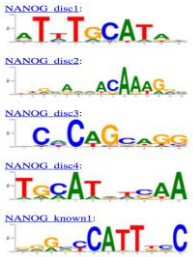


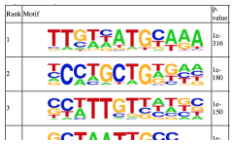


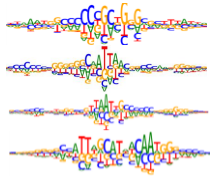
MoDISco patterns outperform traditional motif discovery methods

Logistic regression on top hits to each pattern, auROC









Summary

Locally important patterns in individual examples:

- Perturbation-based methods can find importance scores, but are inefficient and susceptible to saturation
- Backpropagation-based methods are efficient
- Examples of backpropagation methods include deconv nets, gradients, guided-backpropagation, integrated gradients, DeepLIFT
- Most methods make implicit or explicit assumptions about a reference. Having an appropriate reference is important

Globally predictive patterns:

- Visualizing individual filters gives redundant representations
- MoDISco (Motif Discovery From Importance Scores): cluster and aggregate importance scores to find reoccurring patterns

Outstanding issues

- What is the stability of interpretation wrt. different model choices/perturbations to training data/minor perturbations to inputs?
- How to unravel the dependence structure between features learned by the NN?