STANFORD SCHOOL of MEDICINE | Genetics

STANFORD COMPUTER SCIENCE

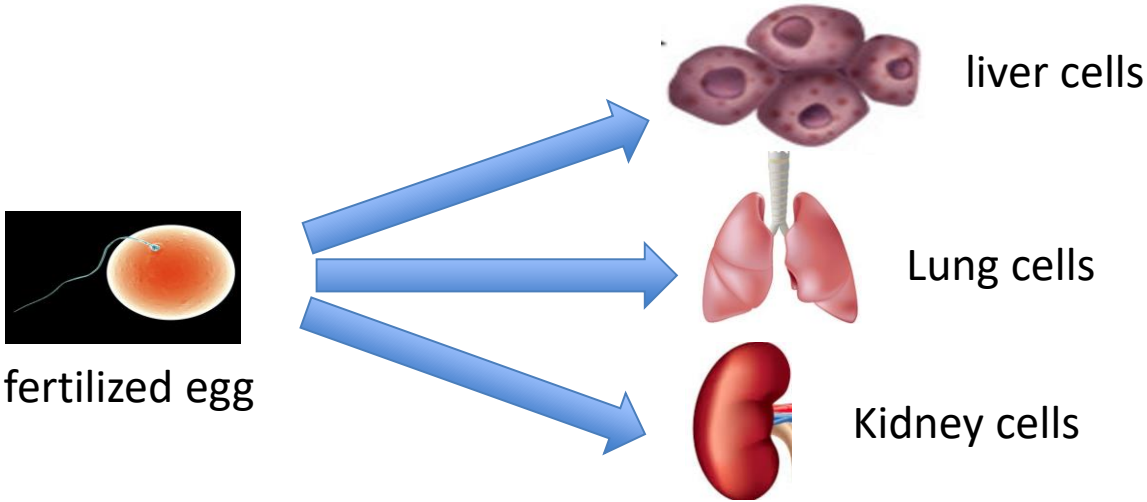
Understanding Genome Regulation with Interpretable Deep Learning

Presented by: Avanti Shrikumar

Kundaje Lab

Stanford University

Example biological problem: understanding stem cell differentiation



Cell-types are different because different genes are turned on

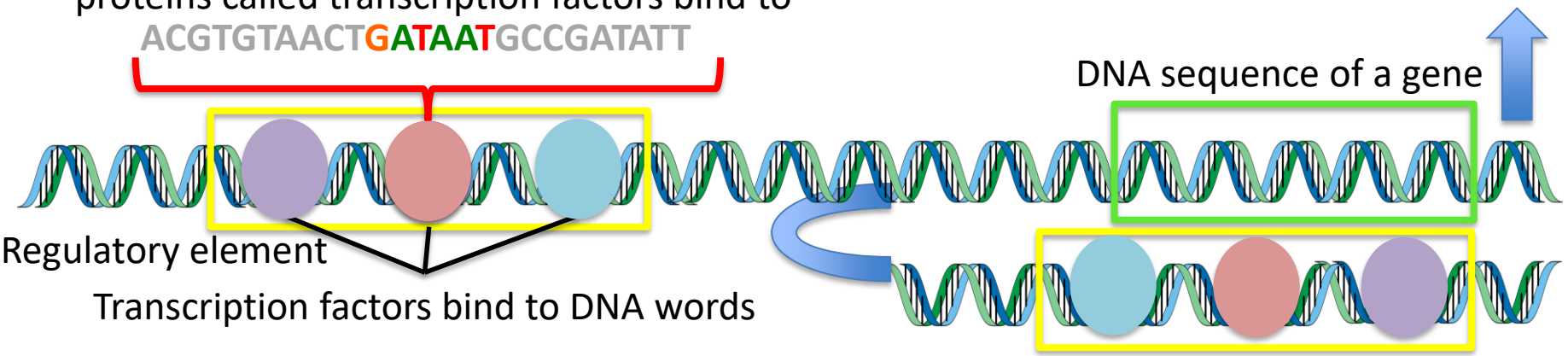
How is cell-type-specific gene expression controlled?

Ans: “regulatory elements” act like switches to turn genes on

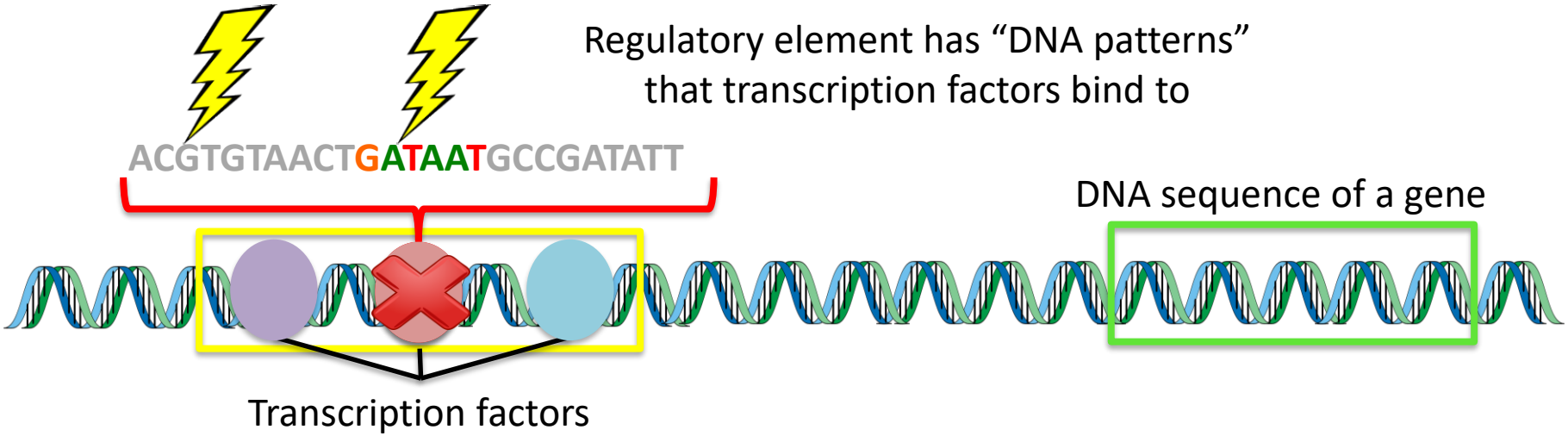
“Regulatory elements” are switches that turn genes on

Sequence contain “DNA patterns” that proteins called transcription factors bind to

ACGTGTAACTGATAATGCCGATATT



90%+* of disease-associated mutations are outside genes!

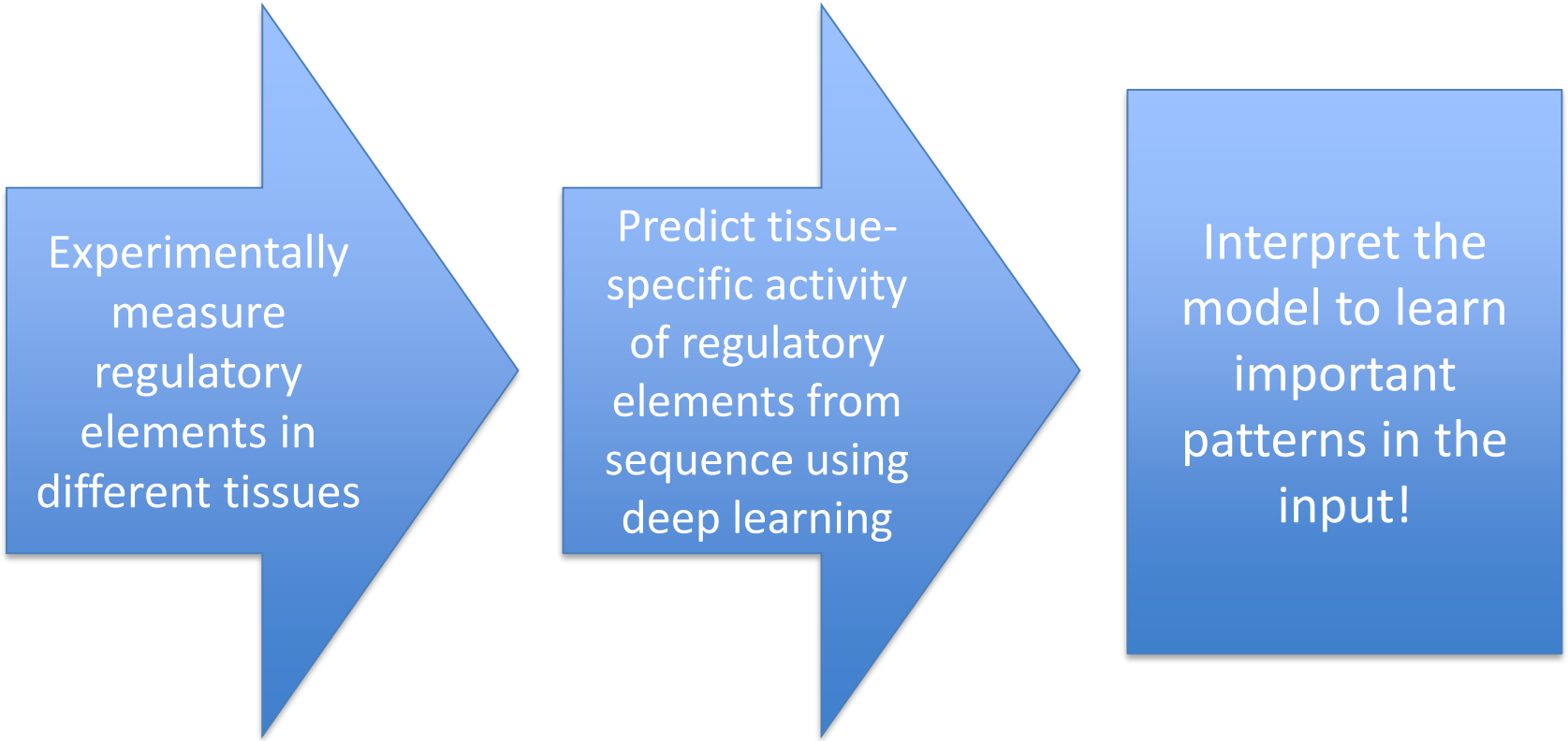


Many positions in a regulatory element are not essential for its function!

→ Which positions in regulatory elements matter?

*Stranger *et al.*, *Genet.*, 2011

Q: Which positions in regulatory elements matter?



Questions for the model

- Which parts of the input are the most important for making a given prediction?
- What are the recurring patterns in the input?

Questions for the model

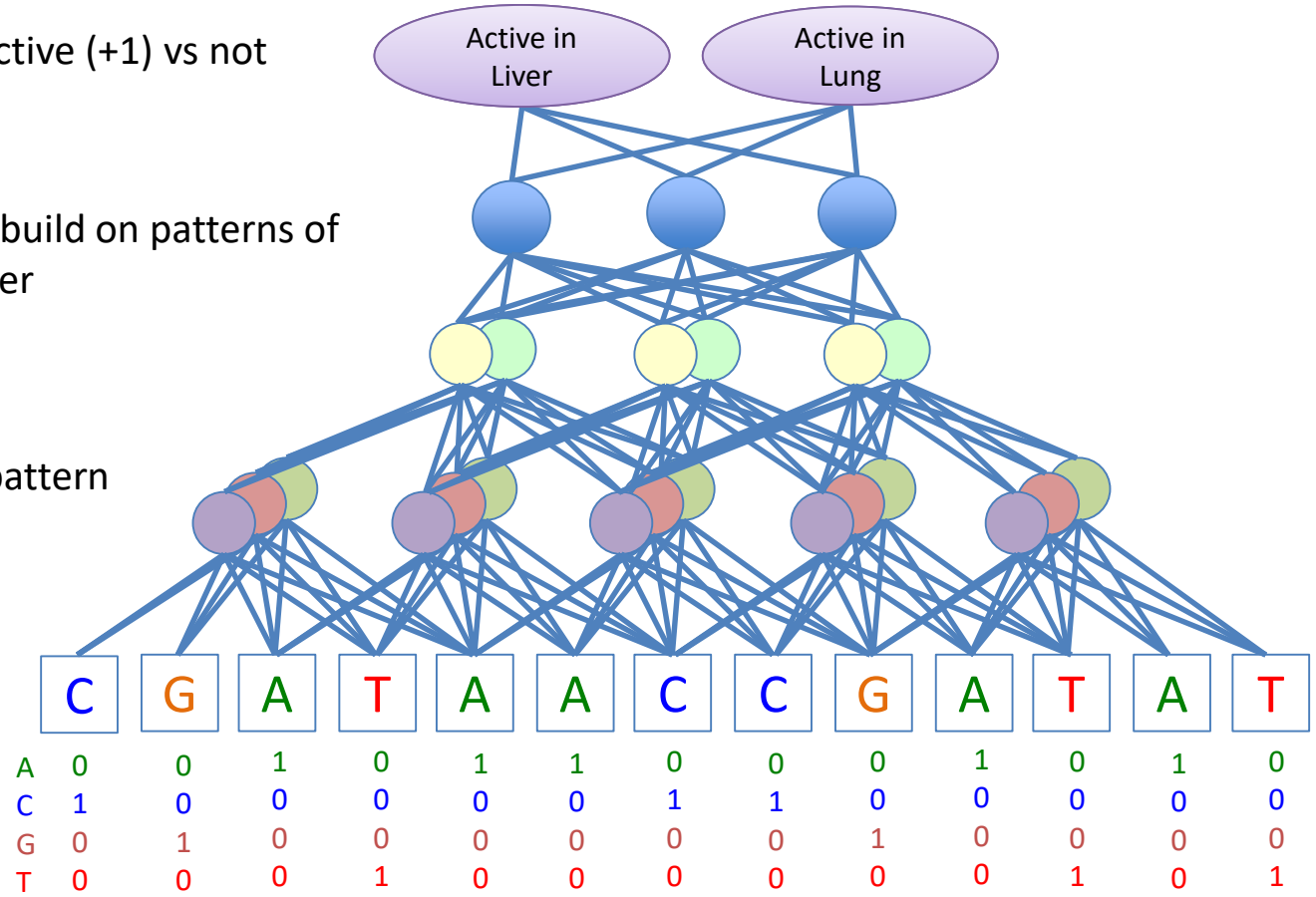
- Which parts of the input are the most important for making a given prediction?
- What are the recurring patterns in the input?

Overview of deep learning model

Output: Active (+1) vs not active (0)

Later layers build on patterns of previous layer

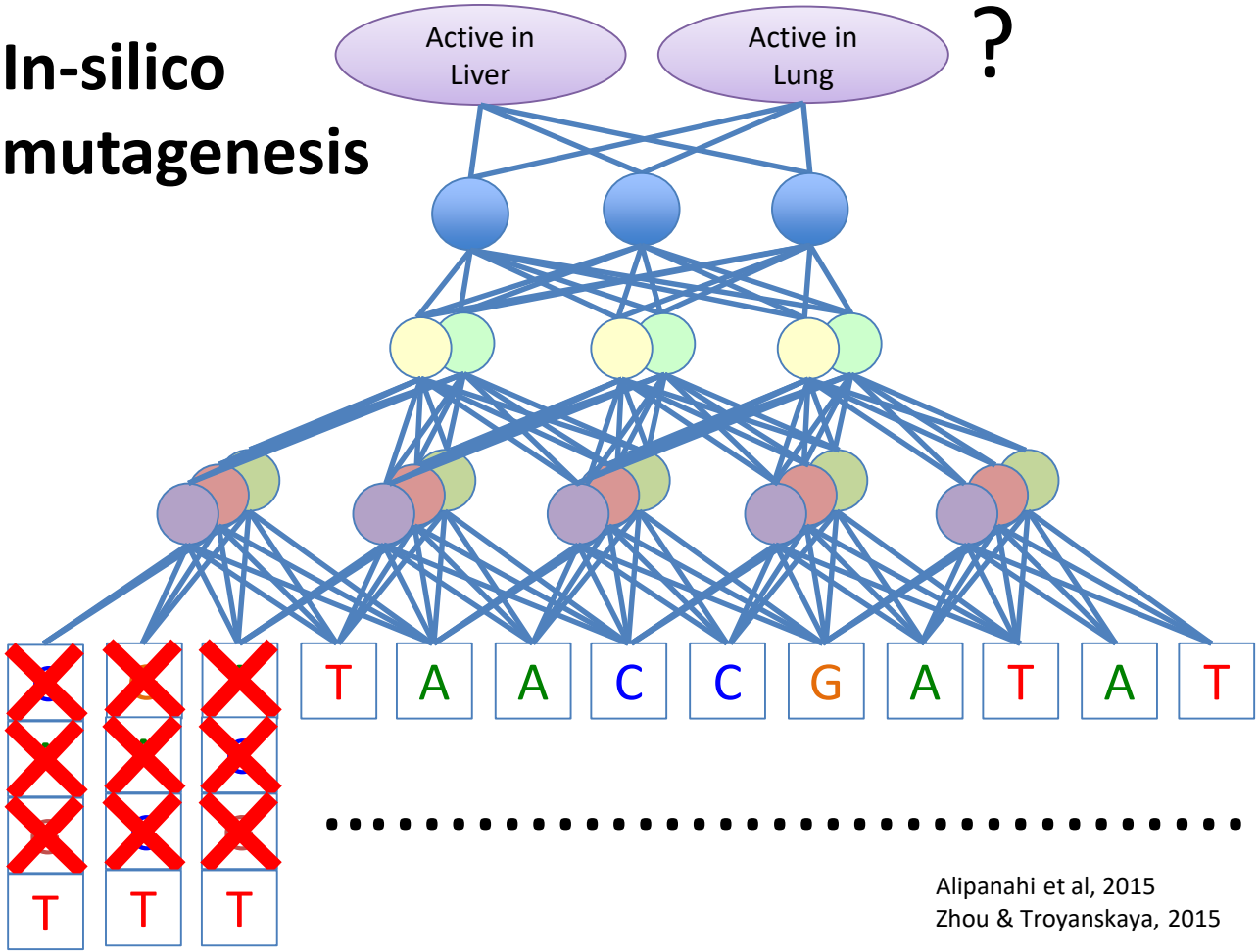
Learned pattern detectors



Input: DNA sequence represented as ones and zeros

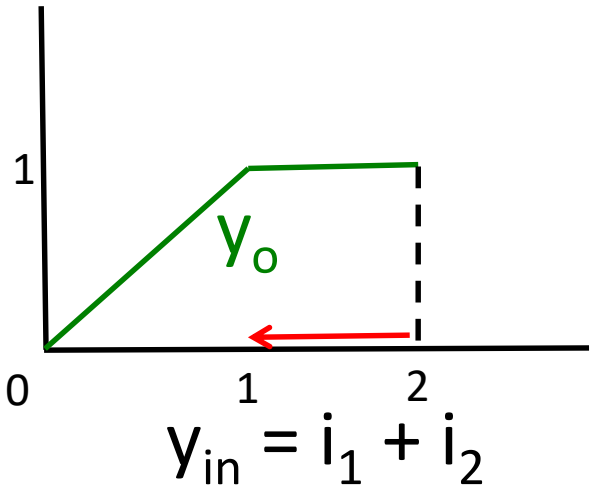
How can we identify important nucleotides?

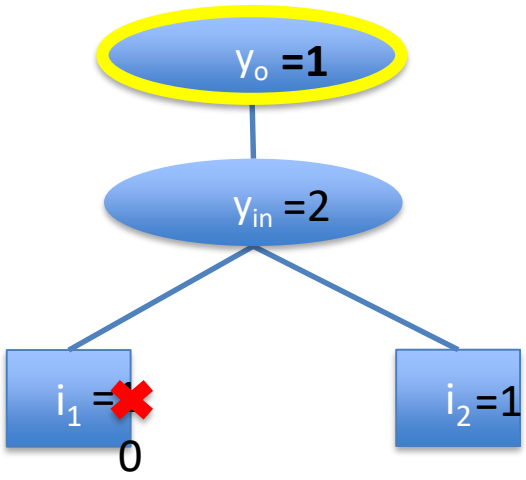
In-silico mutagenesis



Alipanahi et al, 2015
Zhou & Troyanskaya, 2015

Saturation problem illustrated



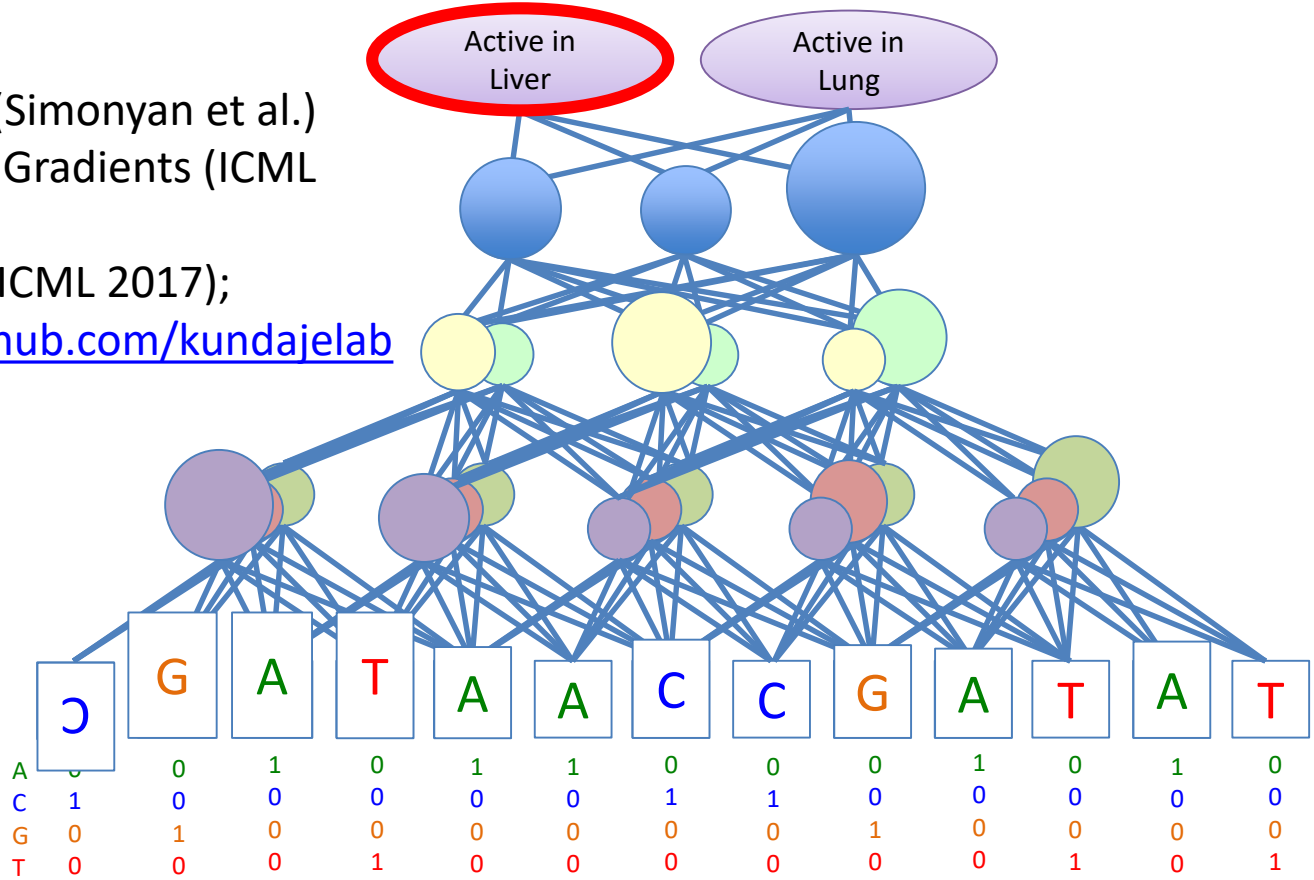


Avoiding saturation means perturbing combinations of inputs → increased computational cost

“Backpropagation” based approaches

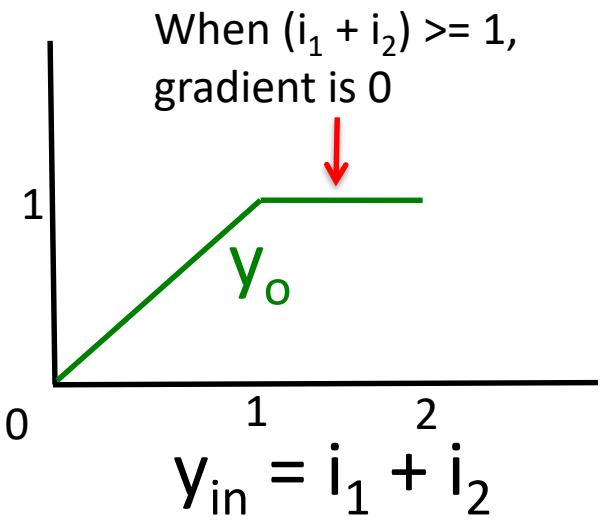
Examples

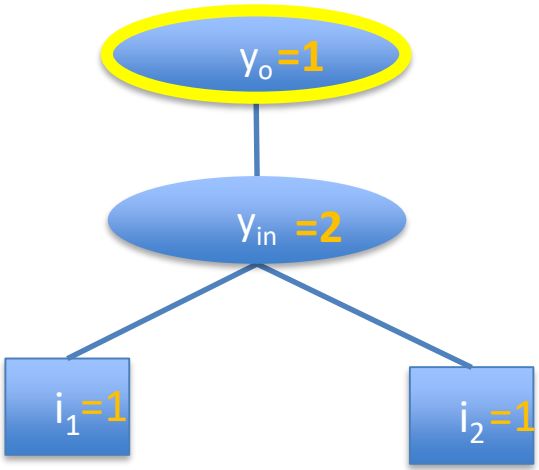
- Gradients (Simonyan et al.)
- Integrated Gradients (ICML 2017)
- DeepLIFT (ICML 2017); https://github.com/kundajelab/deeplift



Input: DNA sequence represented as ones and zeros

Saturation revisited



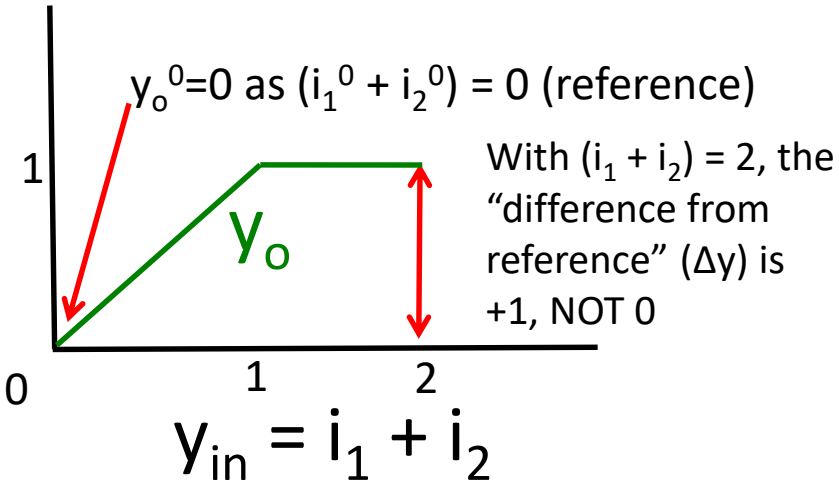


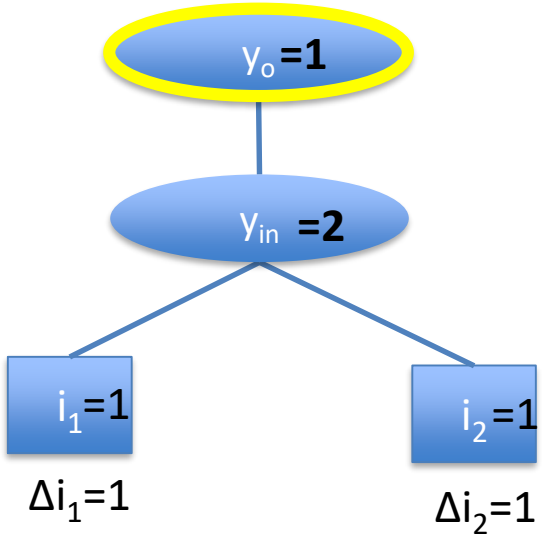
Affects:

- Gradients
- Deconvolutional Networks
- Guided Backpropagation
- Layerwise Relevance Propagation

The DeepLIFT solution: difference from reference

Reference: $i_1^0=0$ & $i_2^0=0$





$C_{\Delta i1\Delta y}=0.5=C_{\Delta i2\Delta y}$

Detailed backpropagation rules in the paper

Anna Shcherbina

DeepLIFT scores at active regulatory element near HNF4A gene

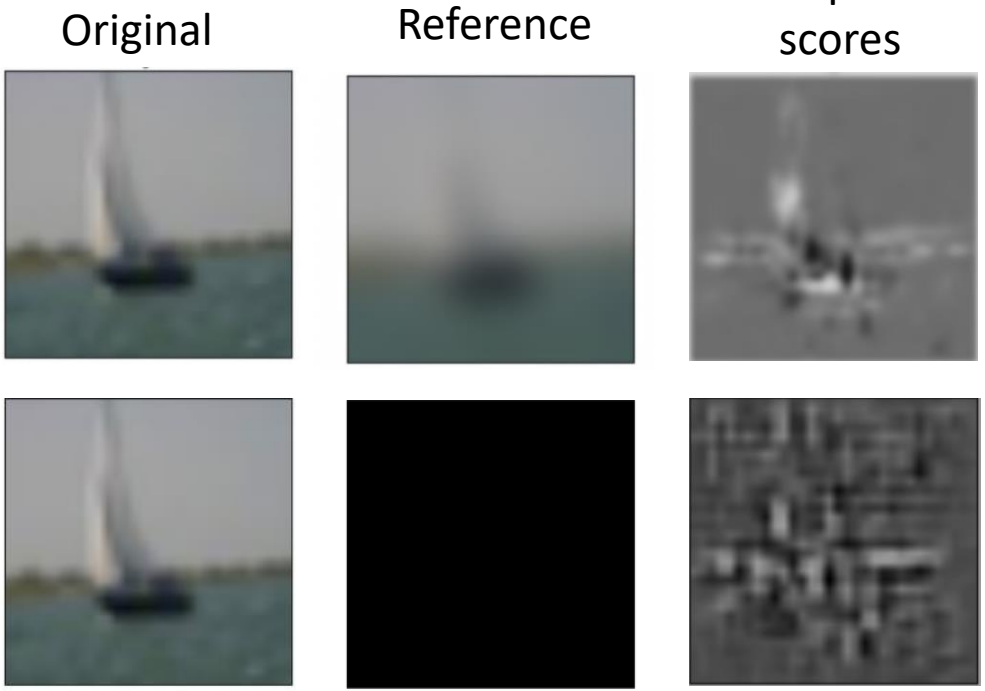
Liver

Lung

Kidney

Choice of reference matters!

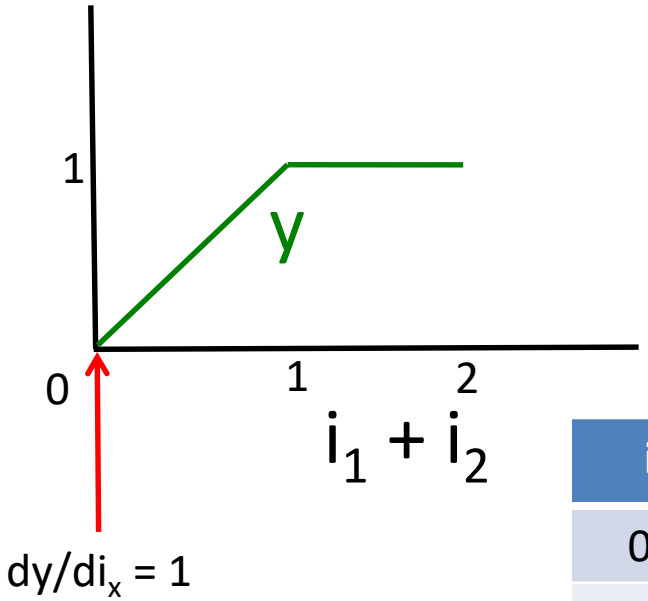
CIFAR10 model, class = “ship”

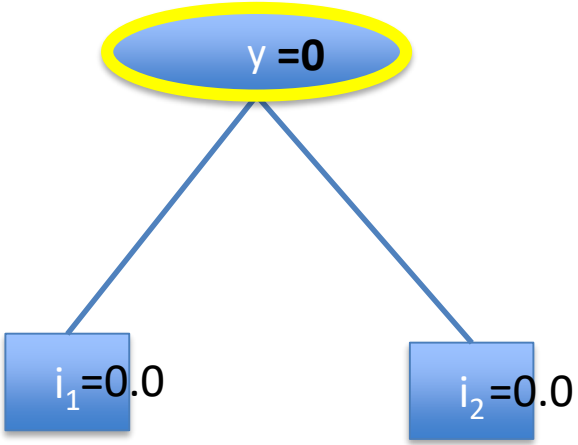


Suggestions on how to pick a reference:

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

Integrated Gradients: Another reference-based approach

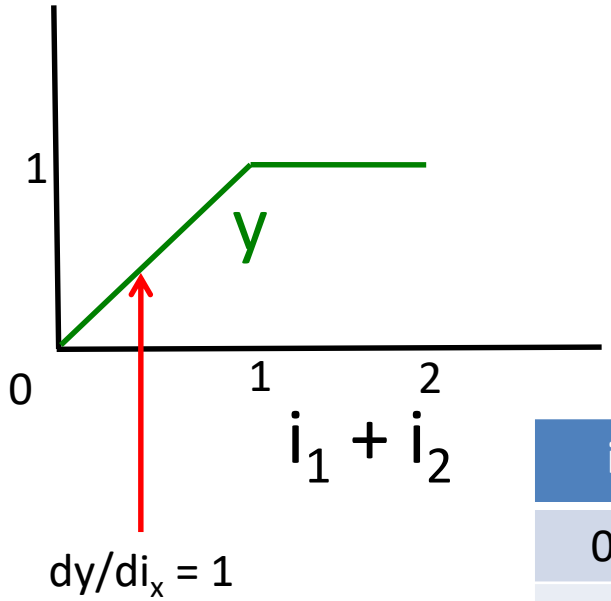


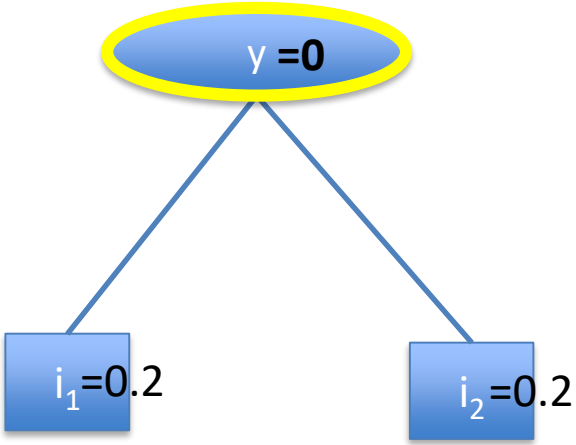


i_1	i_2	dy/di_x
0.0	0.0	1

i_1	i_2	dy/di_x

Integrated Gradients: Another reference-based approach

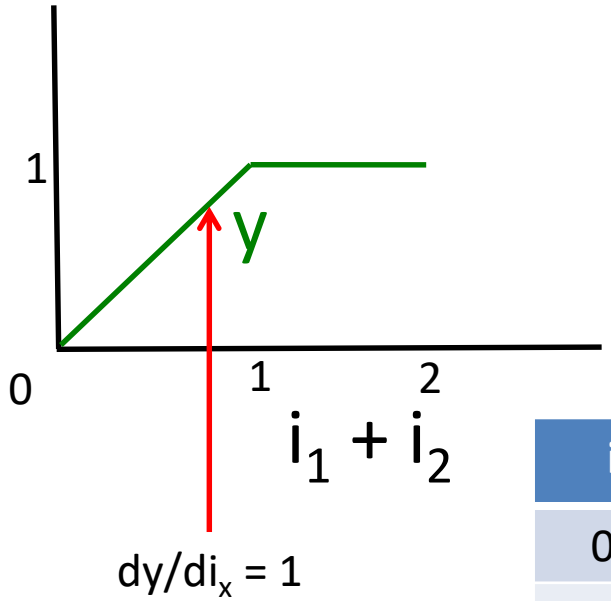


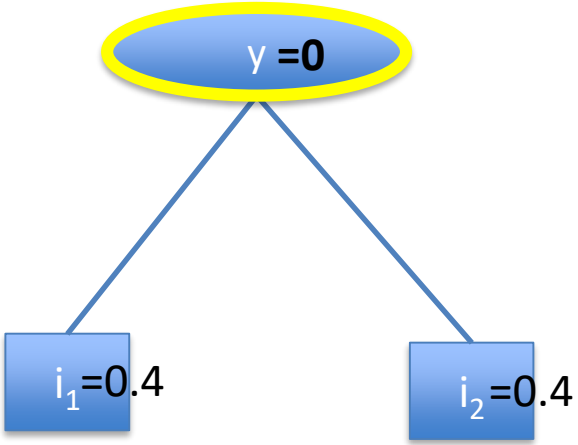


i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1

i_1	i_2	dy/di_x

Integrated Gradients: Another reference-based approach

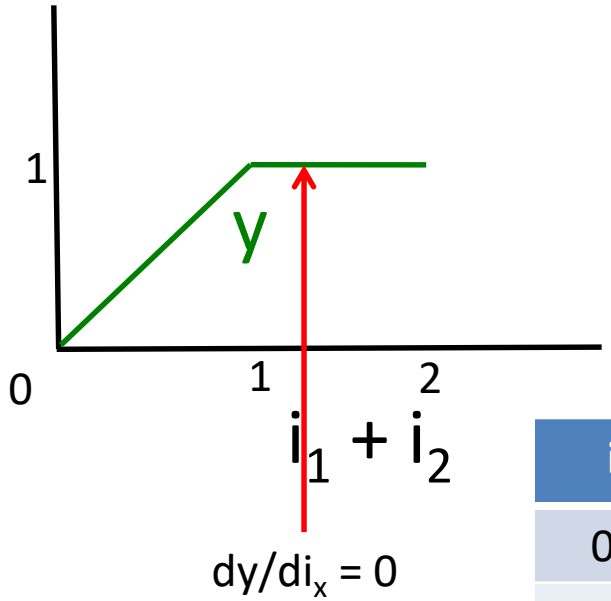


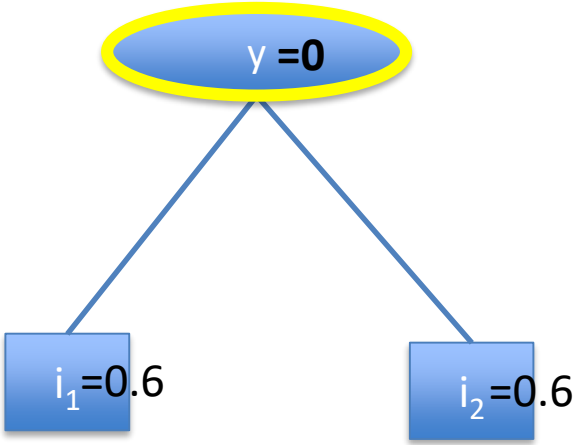


i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1
0.4	0.4	1

i_1	i_2	dy/di_x

Integrated Gradients: Another reference-based approach

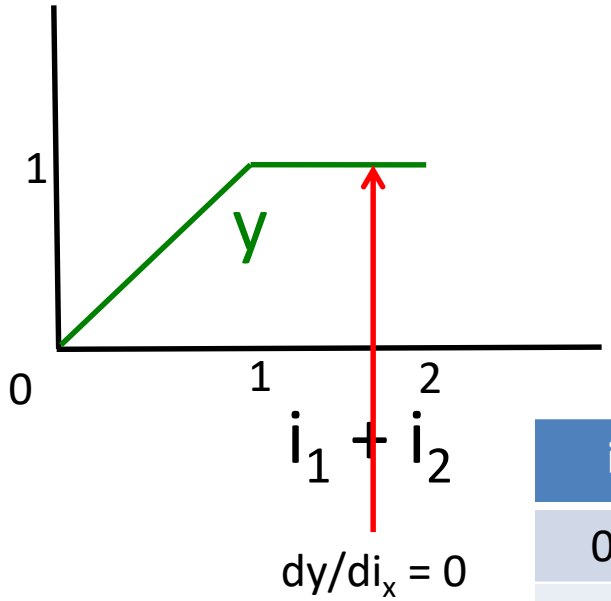


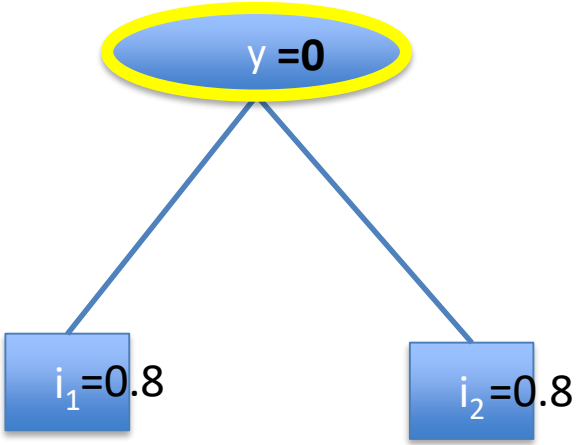


i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1
0.4	0.4	1

i_1	i_2	dy/di_x
0.6	0.6	0

Integrated Gradients: Another reference-based approach

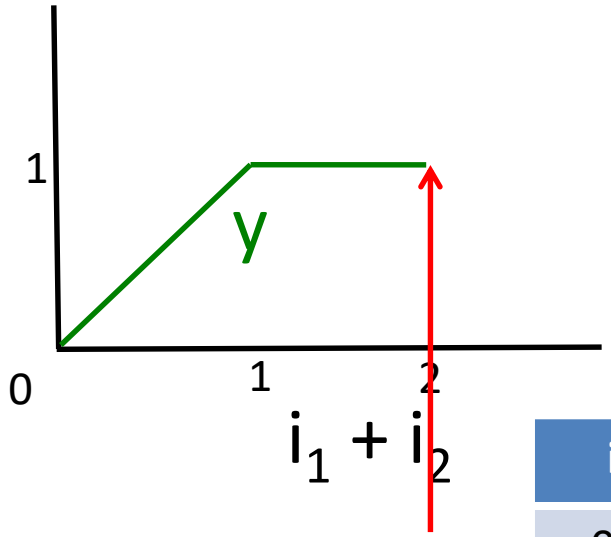


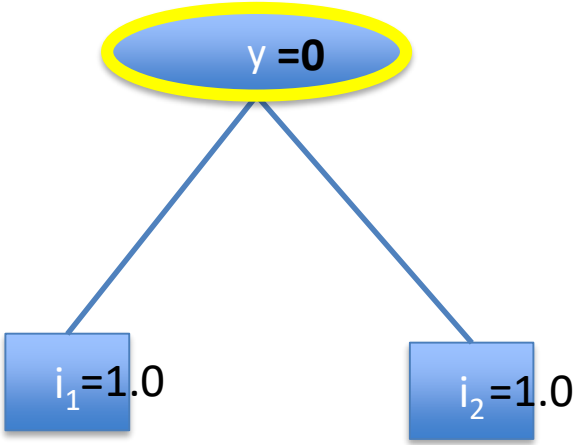


i_1	i_2	dy/di_x
0.0	0.0	1
0.2	0.2	1
0.4	0.4	1

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

Average $dy/di_x = 0.5$

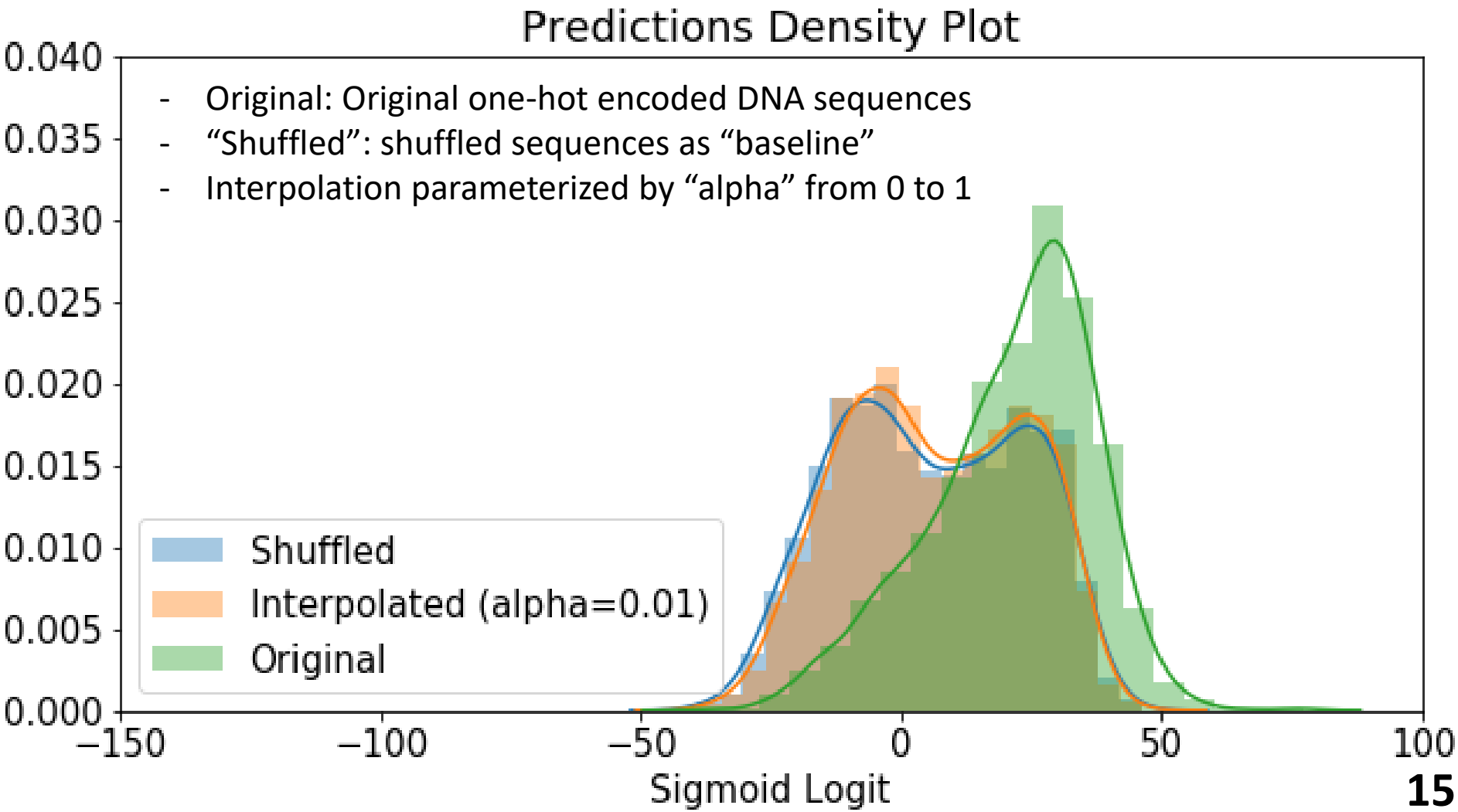
$dy/di_x = 0$

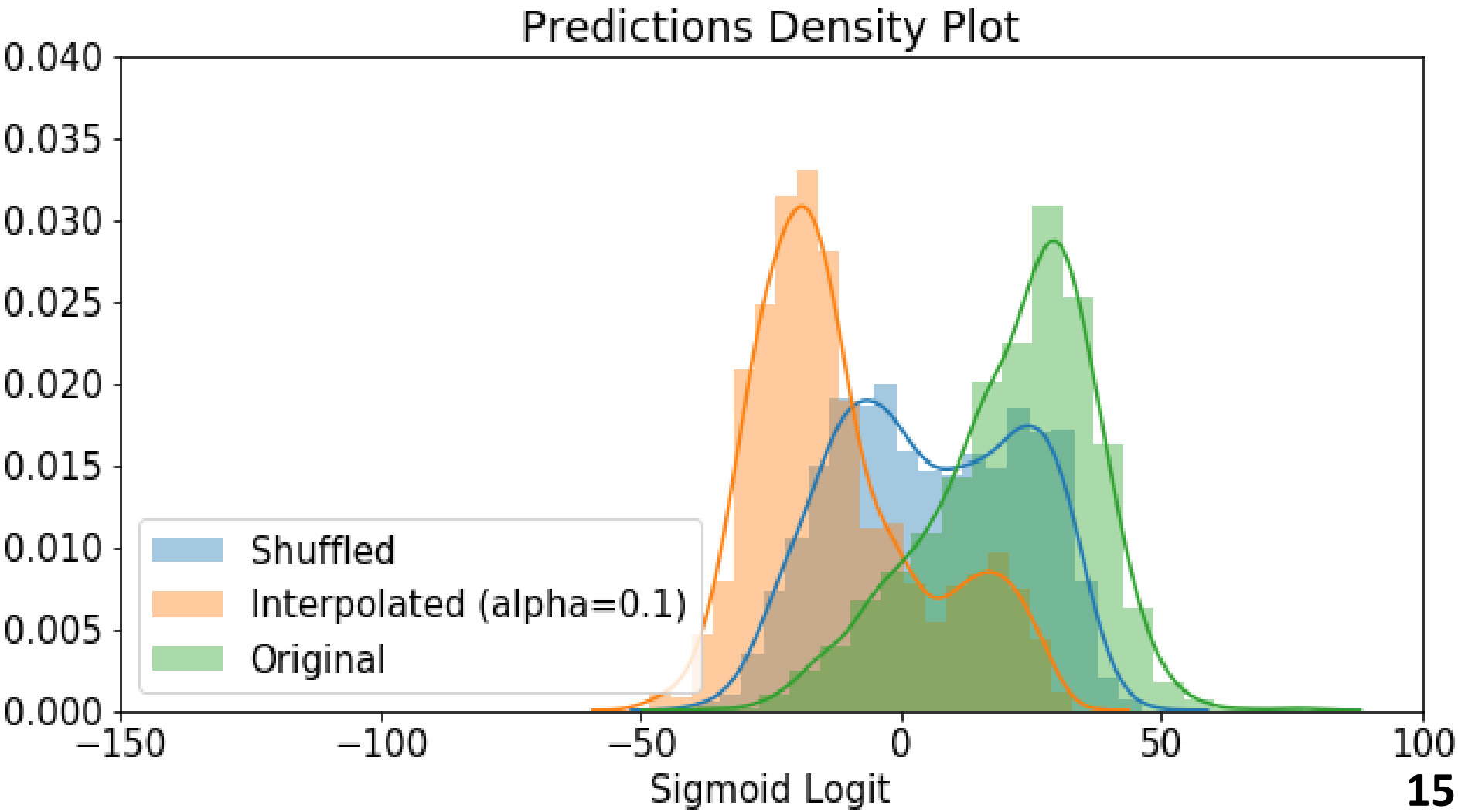
$(\text{Average } dy/di_1)*\Delta i_1 = 0.5$

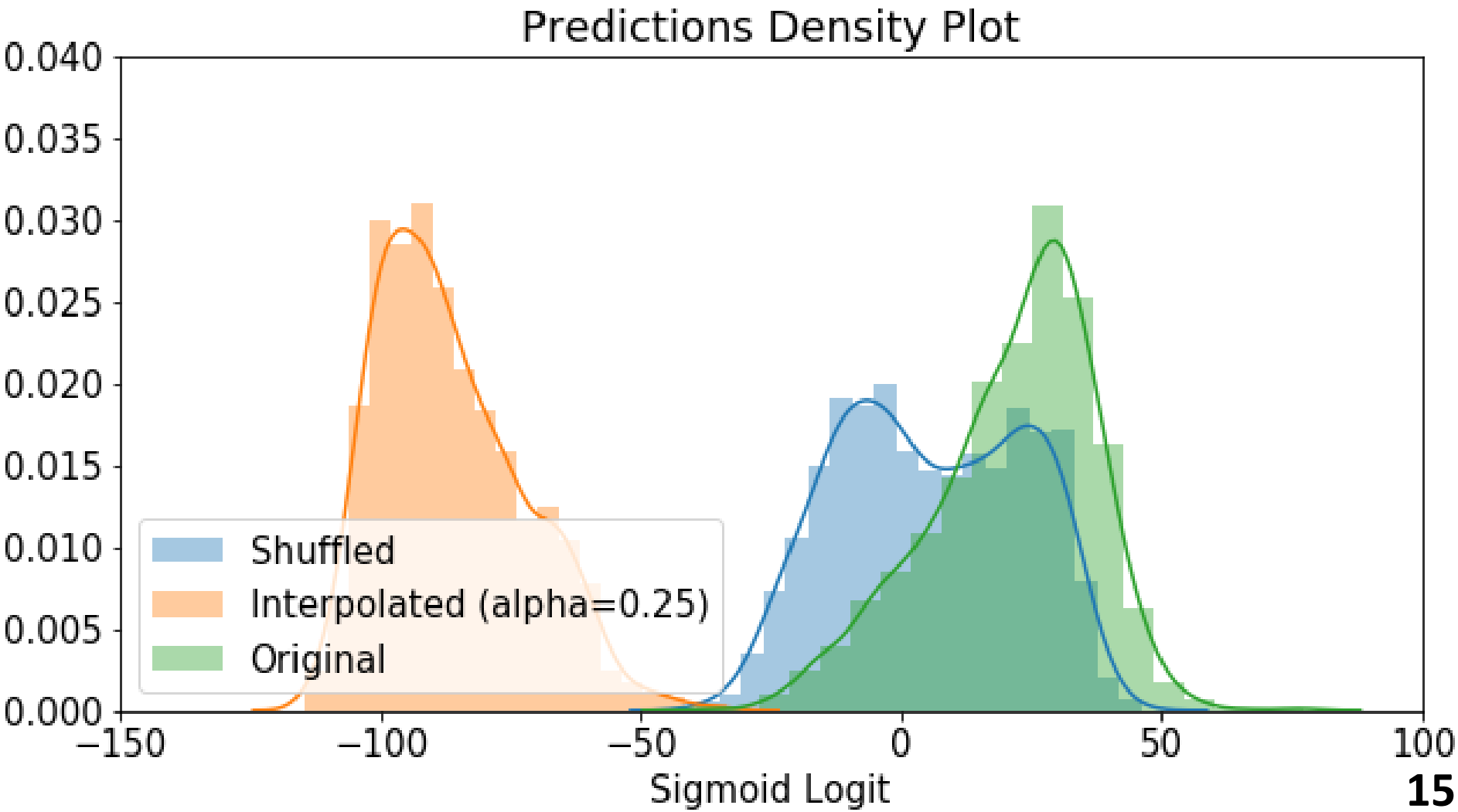
$(\text{Average } dy/di_1)*\Delta i_2 = 0.5$

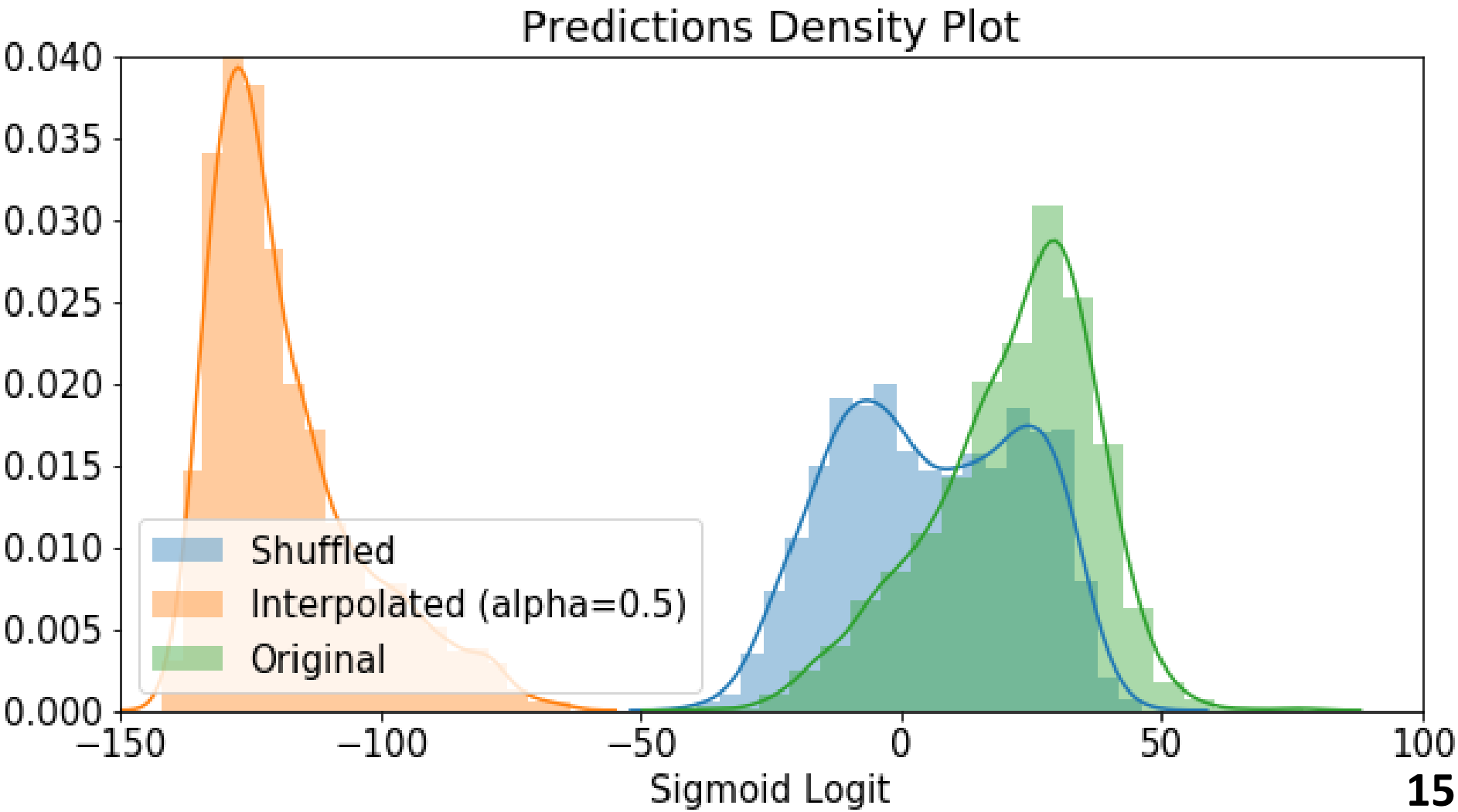
Integrated Gradients: Another reference-based approach

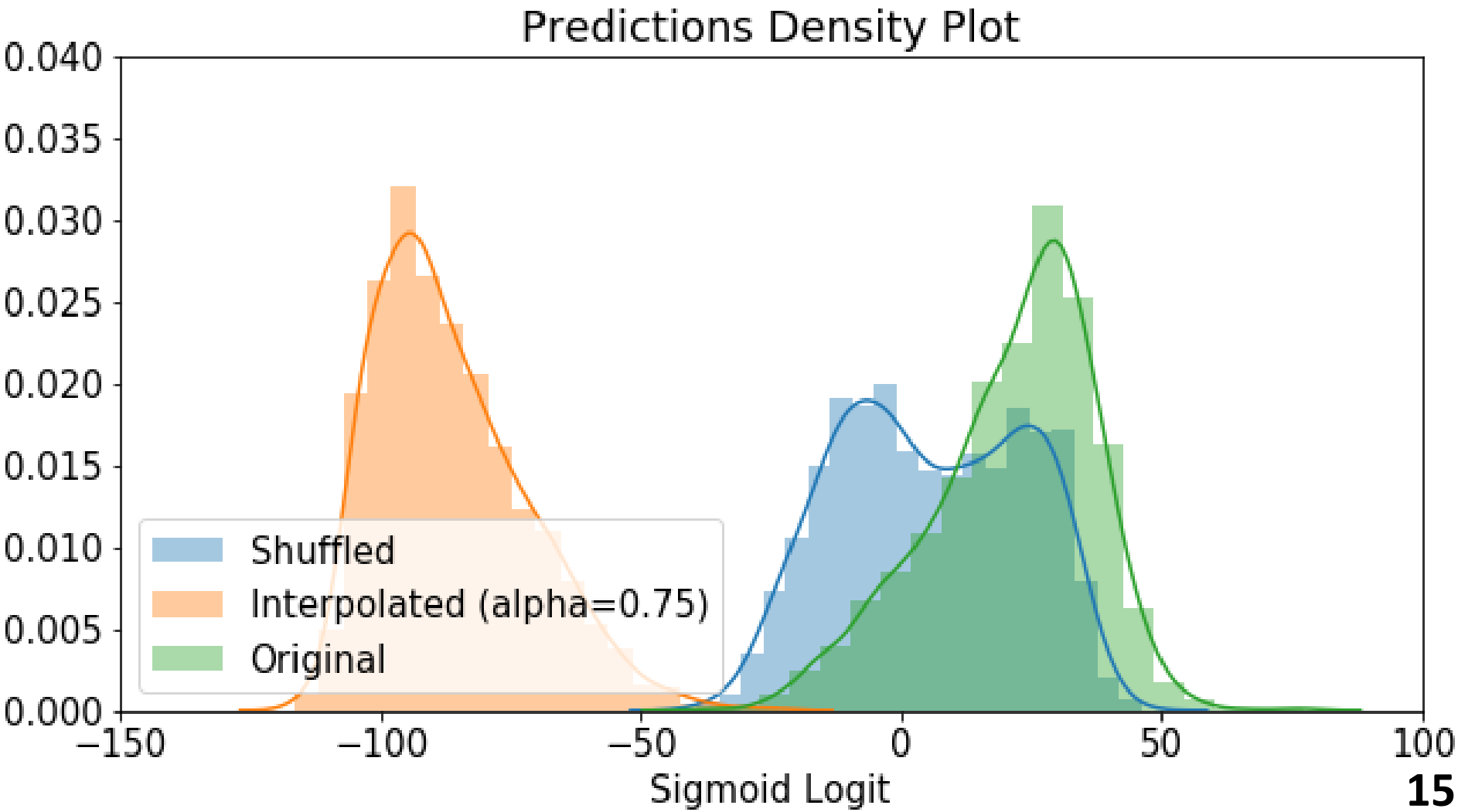
- Sundararajan et al.
- Pros:
  - completely black-box except for gradient computation
  - functionally equivalent networks guaranteed to give the same result
- Cons:
  - Repeated gradient calc. adds computational overhead
  - Linear interpolation path between the baseline and actual input can result in chaotic behavior from the network, esp. for things like one-hot encoded DNA sequence

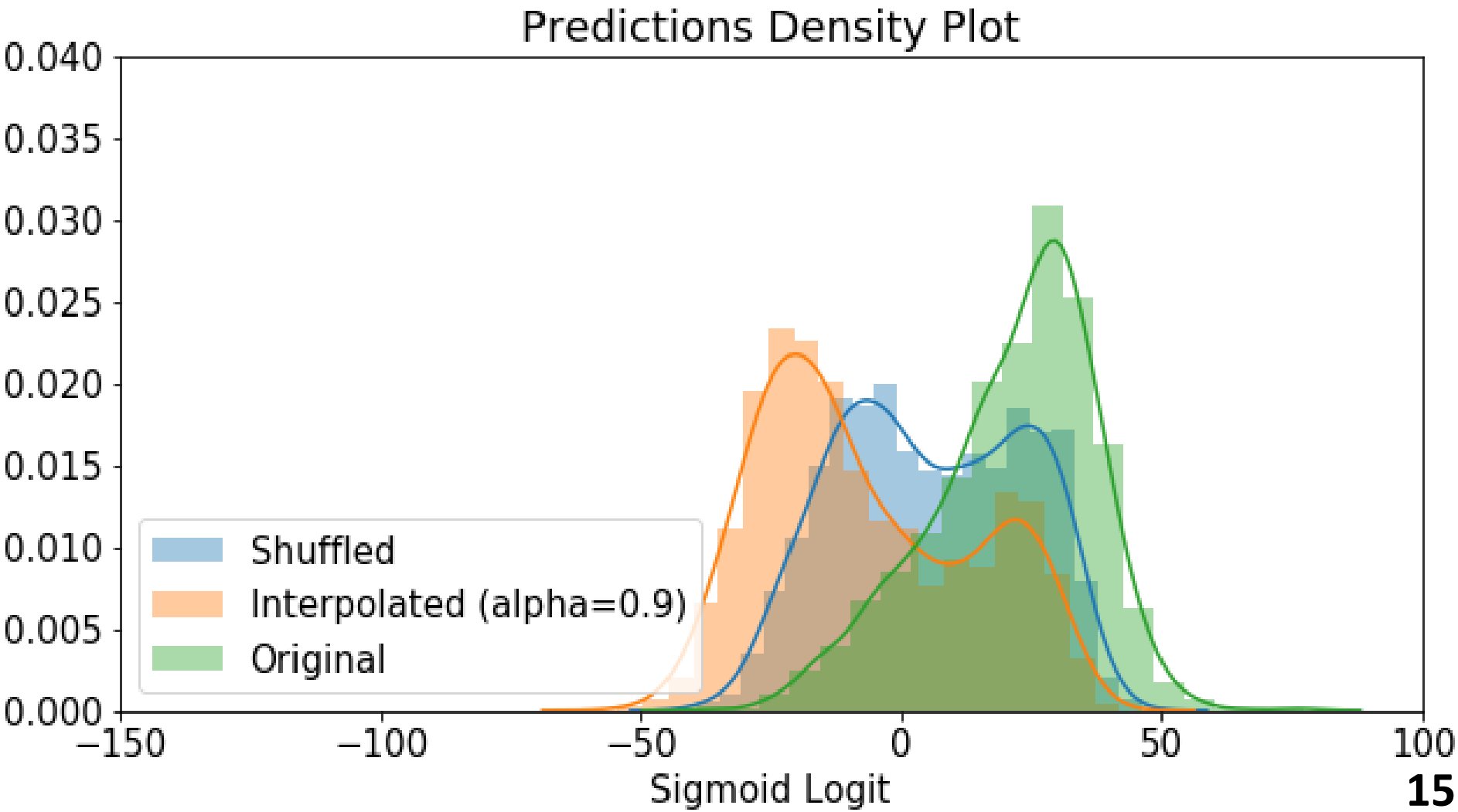

Predictions Density Plot
- Original: Original one-hot encoded DNA sequences
- "Shuffled": shuffled sequences as "baseline"
- Interpolation parameterized by "alpha" from 0 to 1
Shuffled
Interpolated (alpha=0.01)
Original
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−150
−100
−50
0
50
100
Sigmoid Logit

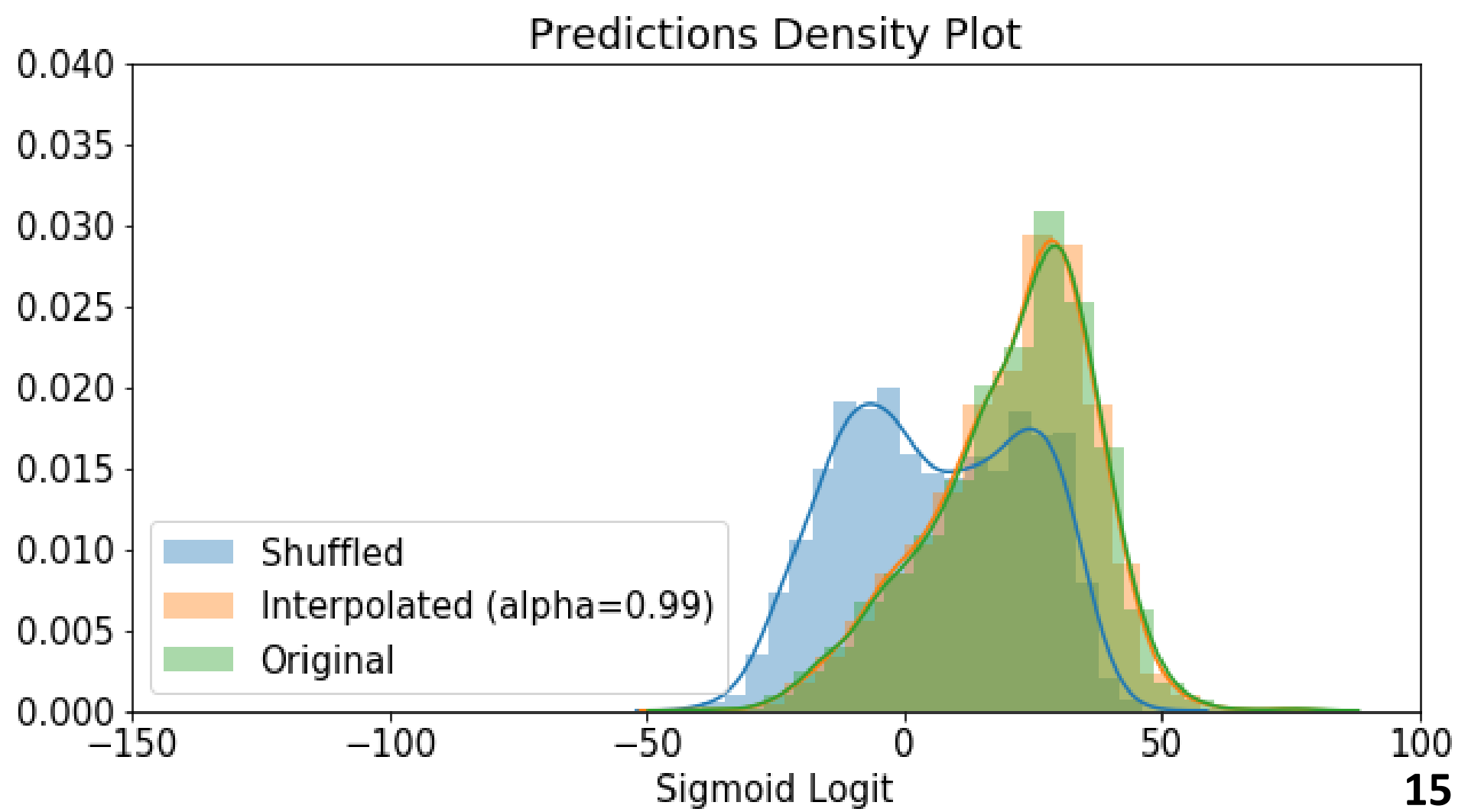

Predictions Density Plot
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−150
−100
−50
0
50
100
Sigmoid Logit
Shuffled
Interpolated (alpha=0.1)
Original

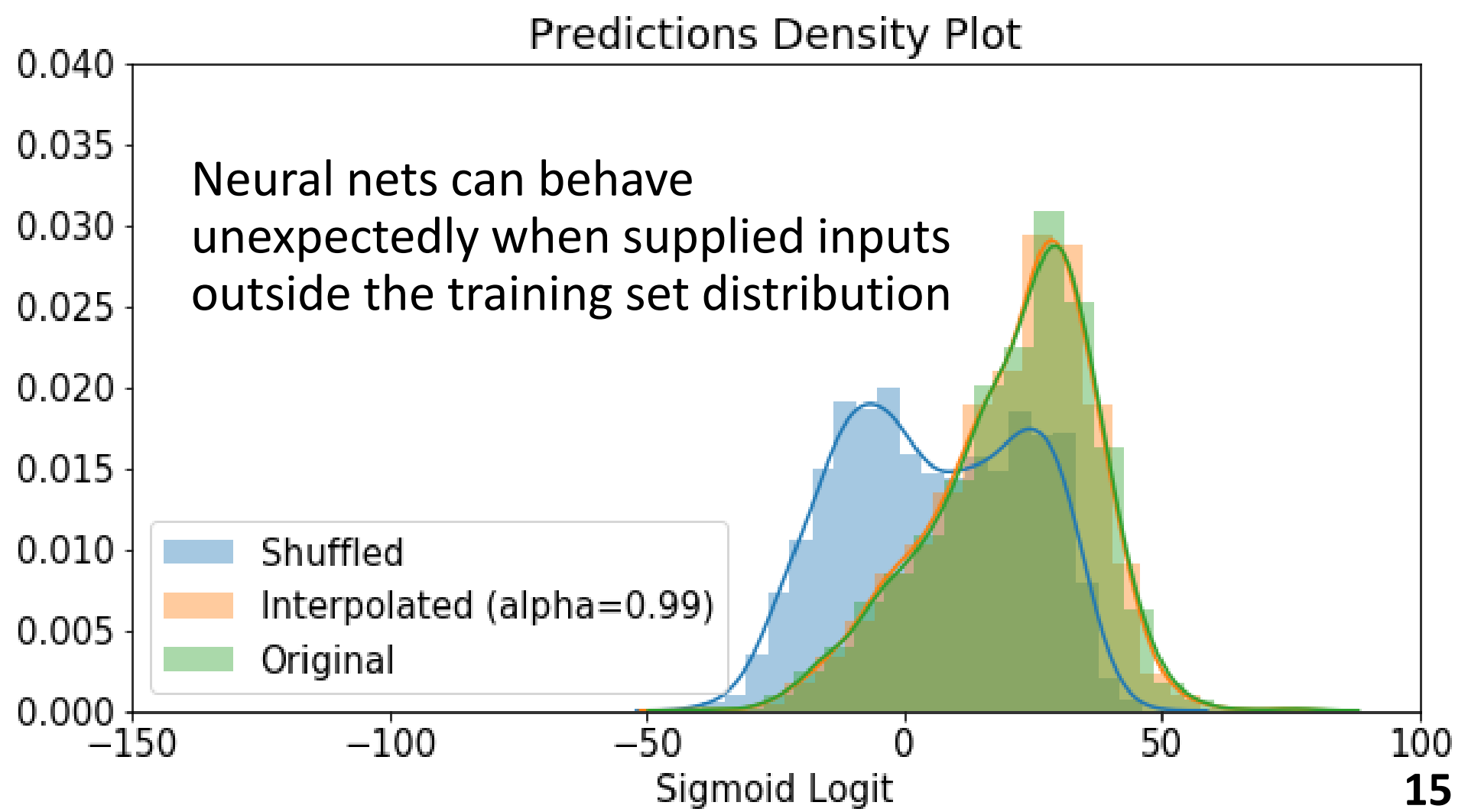

Predictions Density Plot
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−150
−100
−50
0
50
100
Sigmoid Logit
Shuffled
Interpolated (alpha=0.25)
Original


Predictions Density Plot
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−150
−100
−50
0
50
100
Sigmoid Logit
Shuffled
Interpolated (alpha=0.5)
Original


Predictions Density Plot
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−150
−100
−50
0
50
100
Sigmoid Logit
Shuffled
Interpolated (alpha=0.75)
Original


Predictions Density Plot
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−150
−100
−50
0
50
100
Sigmoid Logit
Shuffled
Interpolated (alpha=0.9)
Original


Predictions Density Plot
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−150
−100
−50
0
50
100
Sigmoid Logit
Shuffled
Interpolated (alpha=0.99)
Original

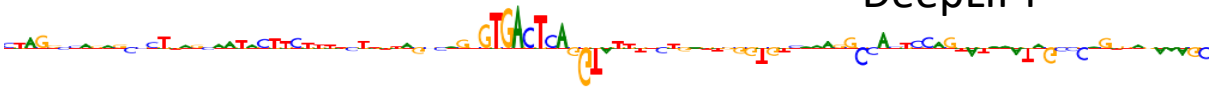

Predictions Density Plot
Neural nets can behave unexpectedly when supplied inputs outside the training set distribution
Shuffled
Interpolated (alpha=0.99)
Original
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−150
−100
−50
0
50
100
Sigmoid Logit

Might be why Integrated Gradients sometimes performs worse than grad*input on DNA…

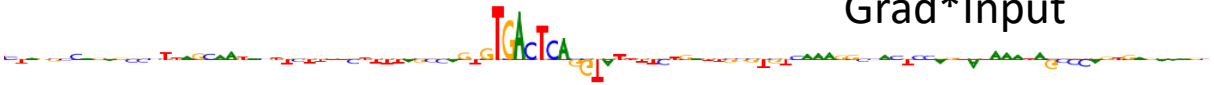
Region active in cell type “A549”

Per-position perturbation
(“In-Silico Mutagenesis”)

DeepLIFT



Grad*Input

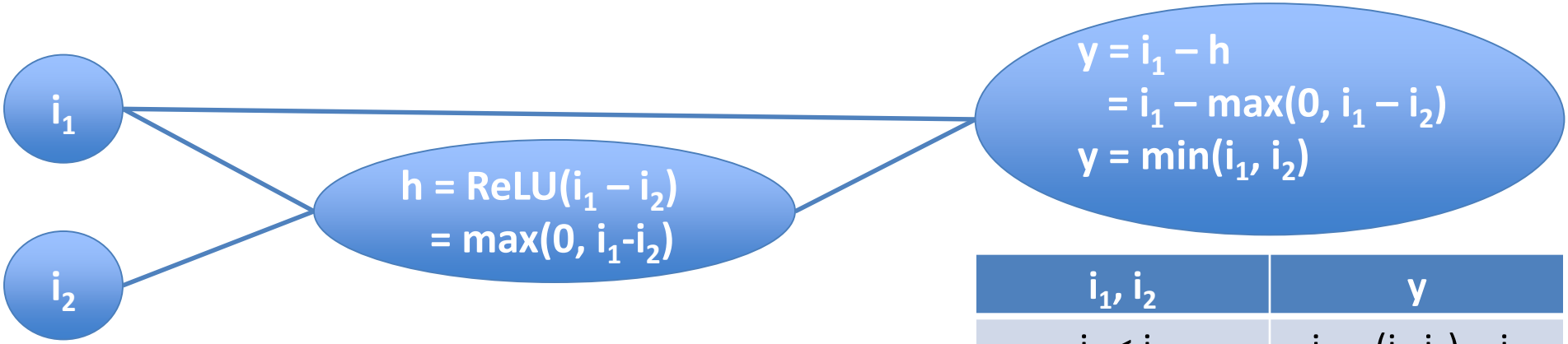


Integrated Gradients

Integrated Gradients: Another reference-based approach

- Sundararajan et al.
- Pros:
  - completely black-box except for gradient computation
  - functionally equivalent networks guaranteed to give the same result
- Cons:
  - Repeated gradient calc. adds computational overhead
  - Linear interpolation path between the baseline and actual input can result in chaotic behavior from the network, esp. for things like one-hot encoded DNA sequence
  - Still relies on gradients, which are local by nature and can give misleading interpretations

Failure-case: “min” (AND) relation



i_1, i_2	y
$i_2 < i_1$	$i_1 - (i_1 - i_2) = i_2$
$i_2 > i_1$	$i_1 - 0 = i_1$

Gradient=0 for either i_1 or i_2, whichever is larger

This is true even when interpolating from (0,0) to (i_1, i_2)!

The DeepLIFT solution: consider different orders for adding positive and negative terms

$i_1 = 10$, $i_2 = 6$

$y = i_1 - \text{ReLU}(i_1 - i_2) = 10 - \text{ReLU}(4) = 6 \leftarrow \min(i_1=10, i_2=6)$

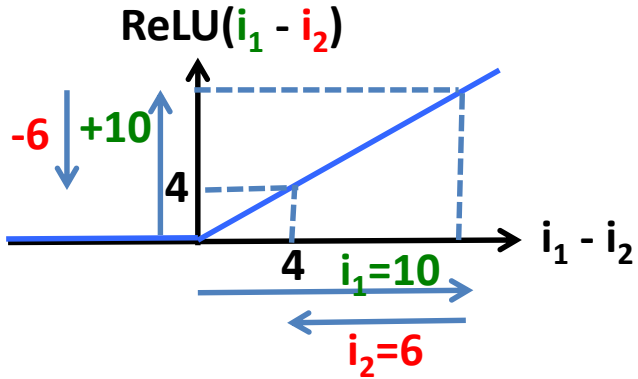
The DeepLIFT solution: consider different orders for adding positive and negative terms

$i_1 = 10, i_2 = 6$

$y = i_1 - \text{ReLU}(i_1 - i_2) = 10 - \text{ReLU}(4) = 6 \leftarrow \min(i_1=10, i_2=6)$

Standard breakdown:

4 = (10 from i_1) + (-6 from i_2)



The DeepLIFT solution: consider different orders for adding positive and negative terms

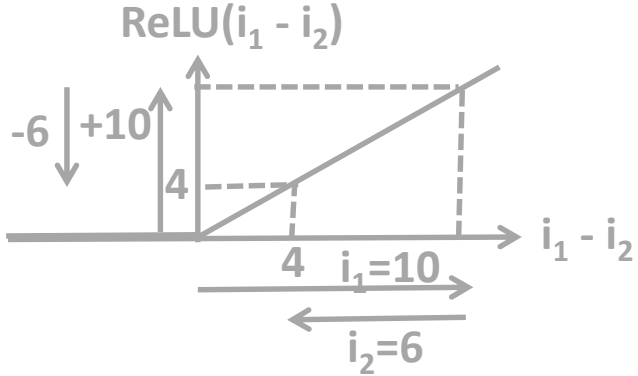
$i_1 = 10, i_2 = 6$

$y = i_1 - \text{ReLU}(i_1 - i_2) = 10 - \text{ReLU}(4) = 6 \leftarrow \min(i_1=10, i_2=6)$

Standard breakdown: $y = 6 =$ (10 from i_1) – [(10 from i_1) – (6 from i_2)] = <u>6 from i_2</u>

Standard breakdown:

4 = (10 from i_1) + (-6 from i_2)



The DeepLIFT solution: consider different orders for adding positive and negative terms

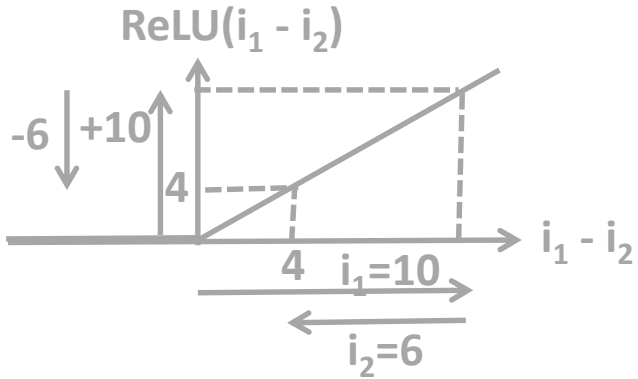
$i_1 = 10, i_2 = 6$

$y = i_1 - \text{ReLU}(i_1 - i_2) = 10 - \text{ReLU}(4) = 6 \leftarrow \min(i_1=10, i_2=6)$

Standard breakdown: $y = 6 = (10 \text{ from } i_1) - [(10 \text{ from } i_1) - (6 \text{ from } i_2)] = \underline{6 \text{ from } i_2}$

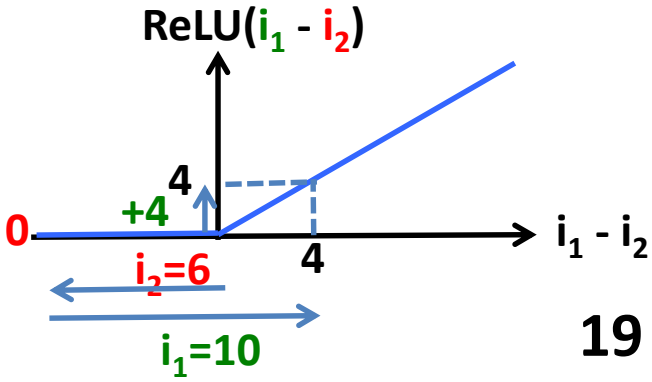
Standard breakdown:

$4 = (10 \text{ from } i_1) + (-6 \text{ from } i_2)$



Other possible breakdown:

$4 = (4 \text{ from } i_1) + (0 \text{ from } i_2)$



The DeepLIFT solution: consider different orders for adding positive and negative terms

$i_1 = 10,\ i_2 = 6$

$y = i_1 - \mathrm{ReLU}(i_1 - i_2) = 10 - \mathrm{ReLU}(4) = 6 \leftarrow \min(i_1{=}10,\ i_2{=}6)$

Standard breakdown: $y = 6 = (10 \text{ from } i_1) - [(10 \text{ from } i_1) - (6 \text{ from } i_2)] = \underline{6 \text{ from } i_2}$

Standard breakdown:

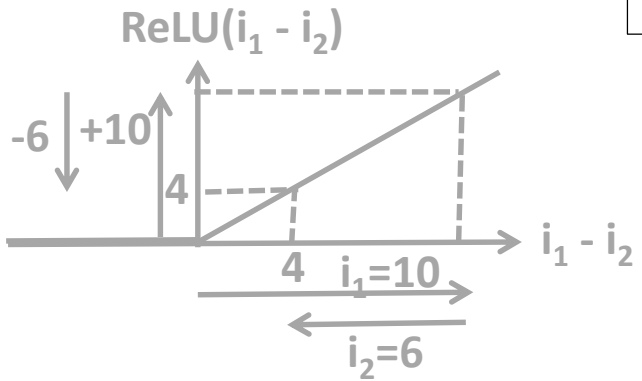
4 = (10 from i_1) + (-6 from i_2) →

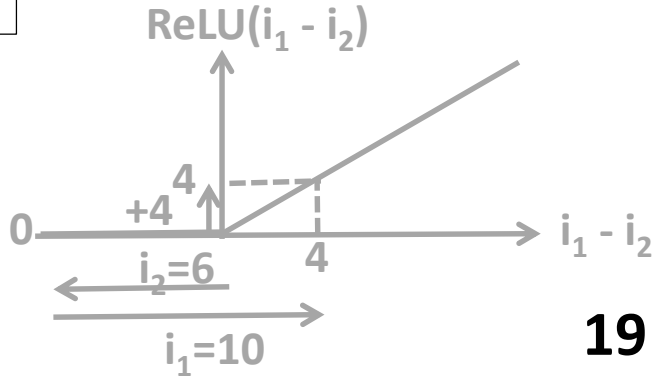
Average i_1 & i_2 contributions:

4 = (7 from i_1) + (-3 from i_2)

← Other possible breakdown:

4 = (4 from i_1) + (0 from i_2)





The DeepLIFT solution: consider different orders for adding positive and negative terms

$i_1 = 10,\ i_2 = 6$

$y = i_1 - \mathrm{ReLU}(i_1 - i_2) = 10 - \mathrm{ReLU}(4) = 6 \leftarrow \min(i_1{=}10,\ i_2{=}6)$

Standard breakdown: $y = 6 = (10 \text{ from } i_1) - [(10 \text{ from } i_1) - (6 \text{ from } i_2)] = \underline{6 \text{ from } i_2}$

Average over both orders: $y = 6 = (10 \text{ from } i_1) - [(7 \text{ from } i_1) + (-3 \text{ from } i_2)]$

$= \underline{(3 \text{ from } i_1) + (3 \text{ from } i_2)}$

Standard breakdown:

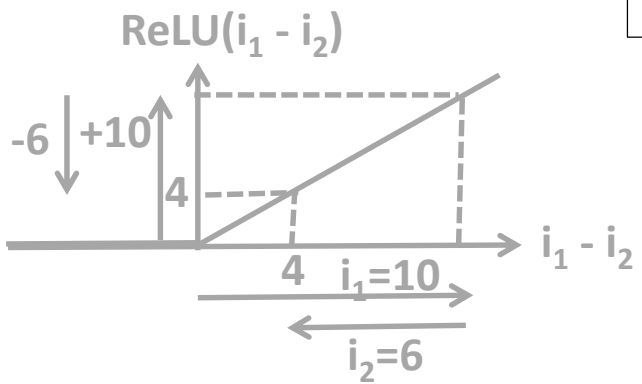
$4 = (10 \text{ from } i_1) + (-6 \text{ from } i_2)$ →

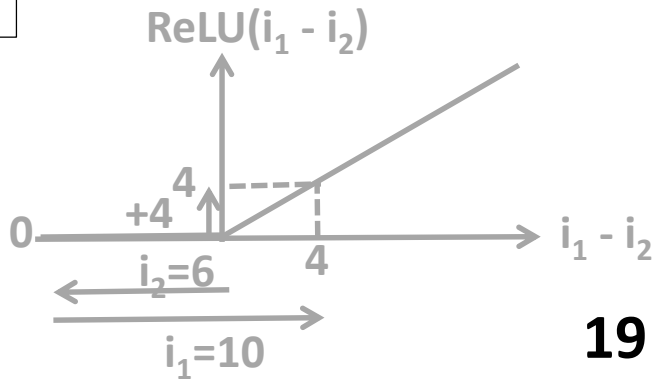
Average i_1 & i_2 contributions:

$4 = (7 \text{ from } i_1) + (-3 \text{ from } i_2)$

← Other possible breakdown:

$4 = (4 \text{ from } i_1) + (0 \text{ from } i_2)$





The DeepLIFT solution: consider different orders for adding positive and negative terms

$i_1 = 10,\ i_2 = 6$

$y = i_1 - \mathrm{ReLU}(i_1 - i_2) = 10 - \mathrm{ReLU}(4) = 6 \leftarrow \min(i_1{=}10,\ i_2{=}6)$

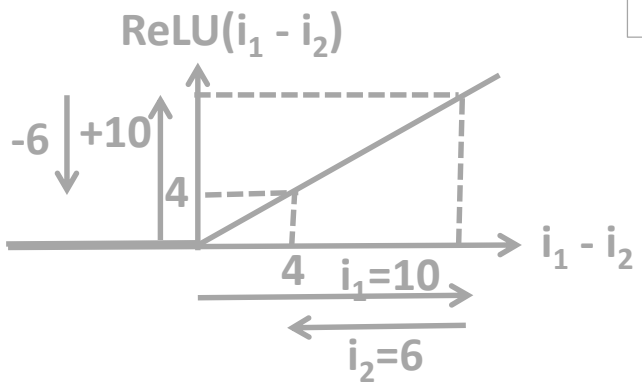
Standard breakdown: $y = 6 = (10 \text{ from } i_1) - [(10 \text{ from } i_1) - (6 \text{ from } i_2)] = \underline{6 \text{ from } i_2}$

Average over both orders: $y = 6 = (10 \text{ from } i_1) - [(7 \text{ from } i_1) + (-3 \text{ from } i_2)]$

$= \underline{(3 \text{ from } i_1) + (3 \text{ from } i_2)}$

Standard breakdown:

$4 = (10 \text{ from } i_1) + (-6 \text{ from } i_2)$



Average i_1 & i_2 contributions:

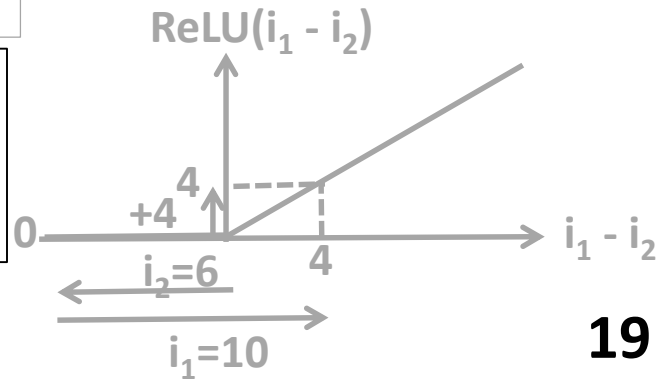
$4 = (7 \text{ from } i_1) + (-3 \text{ from } i_2)$

> 2 inputs: club pos & neg inputs into 2 "meta" terms, assign importance, distribute proportionally

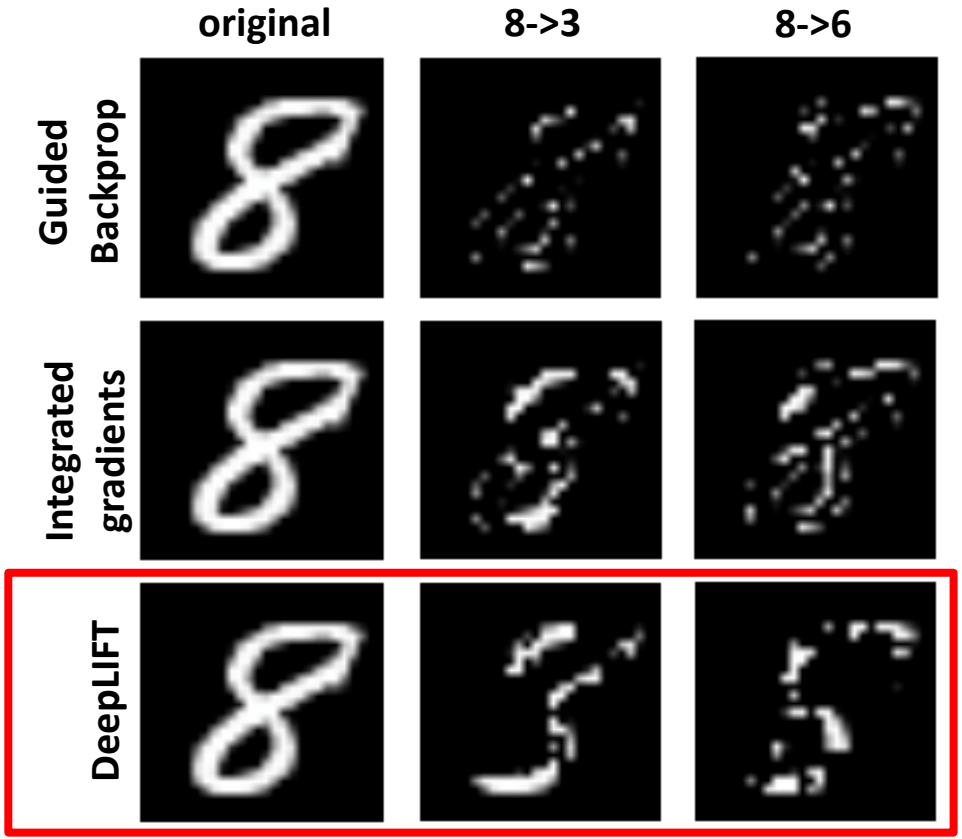
"A unified approach to interpreting model predictions" - Lundberg & Lee

Other possible breakdown:

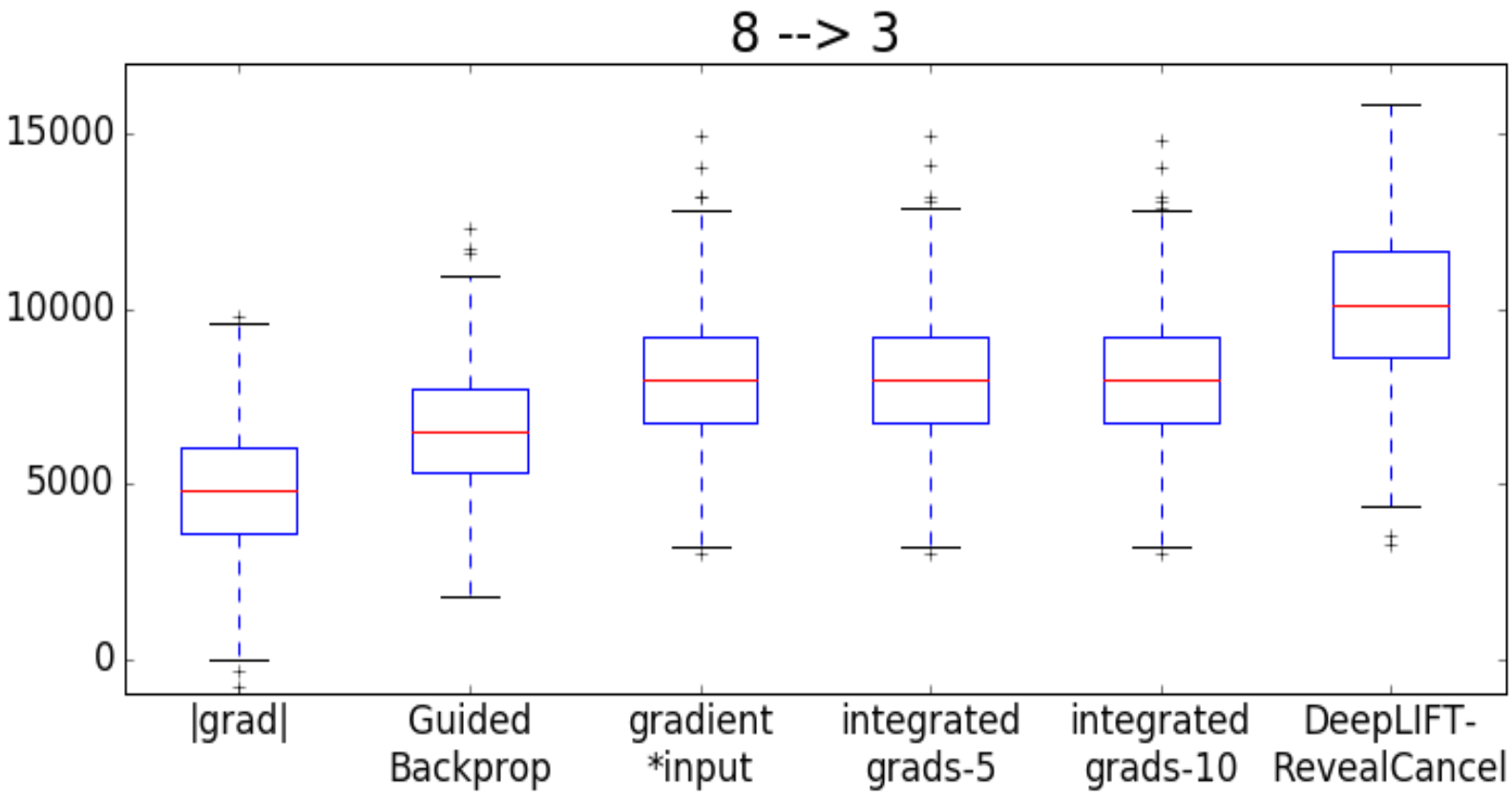
$4 = (4 \text{ from } i_1) + (0 \text{ from } i_2)$



Eg: morphing 8 to a 3 or a 6



Change in log-odds after morphing



What do we gain (in terms of biology knowledge) from using Deep Learning?

Conventional models of protein binding explain only a small fraction of regulatory genetic variants

Pooled ChIP-Seq Links Variation in Transcription Factor Binding to Complex Disease Risk **Cell**

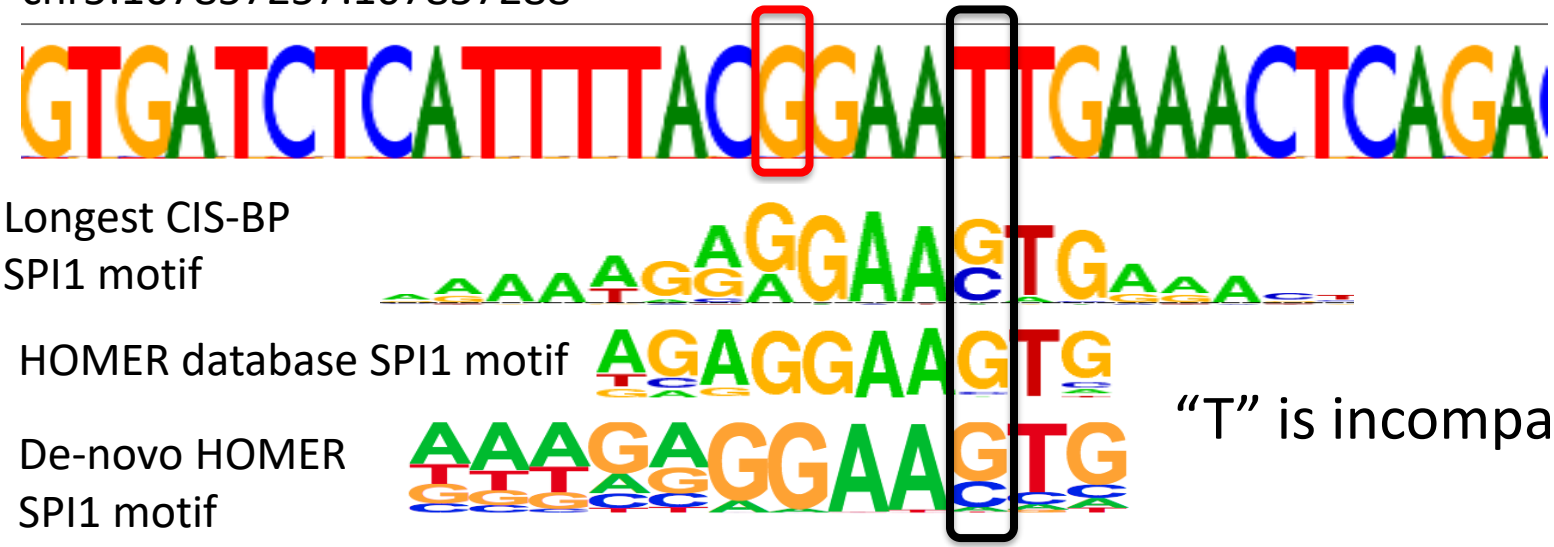
Ashley K. Tehranchi • Marsha Myrthil • Trevor Martin • Brian L. Hie • David Golan • Hunter B. Fraser

For all five DNA-binding proteins studied, **less than 0.9%** of genetic variants affecting binding were located in known patterns (“motifs”)

Example genetic variant affecting binding that is “outside a known motif”

Genetic variant affecting SPI1 binding (p value: 1.6E-6)

chr5:107857257:107857288



“T” is incompatible

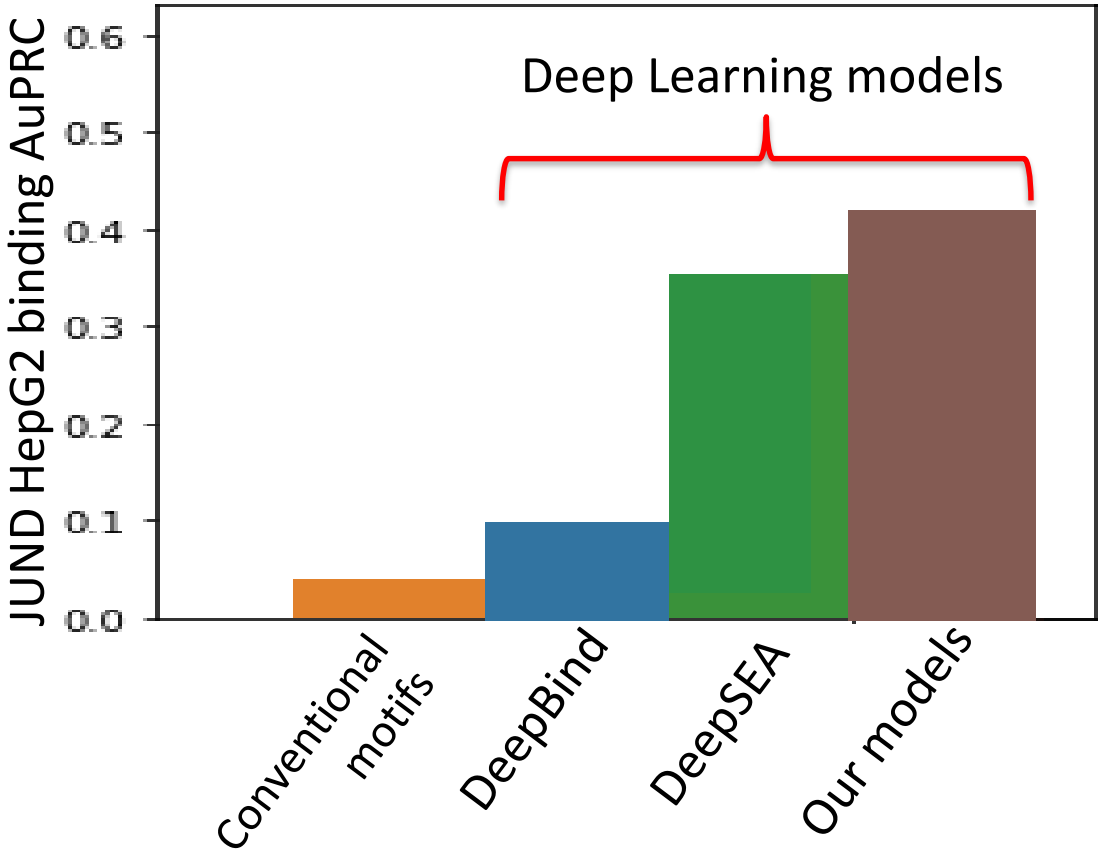
Conventional motifs are too simplified!

Protein–DNA binding: complexities and multi-protein codes

Trevor Siggers ✉, Raluca Gordân

simple protein–DNA recognition codes do not exist.

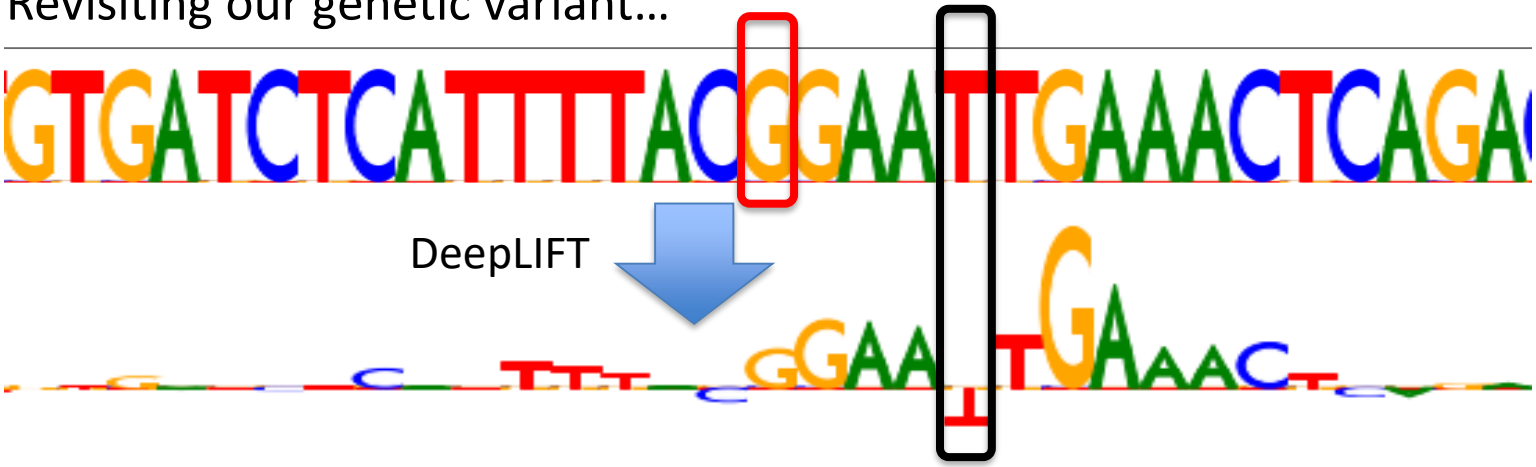
Deep Learning far outperforms PWMs…

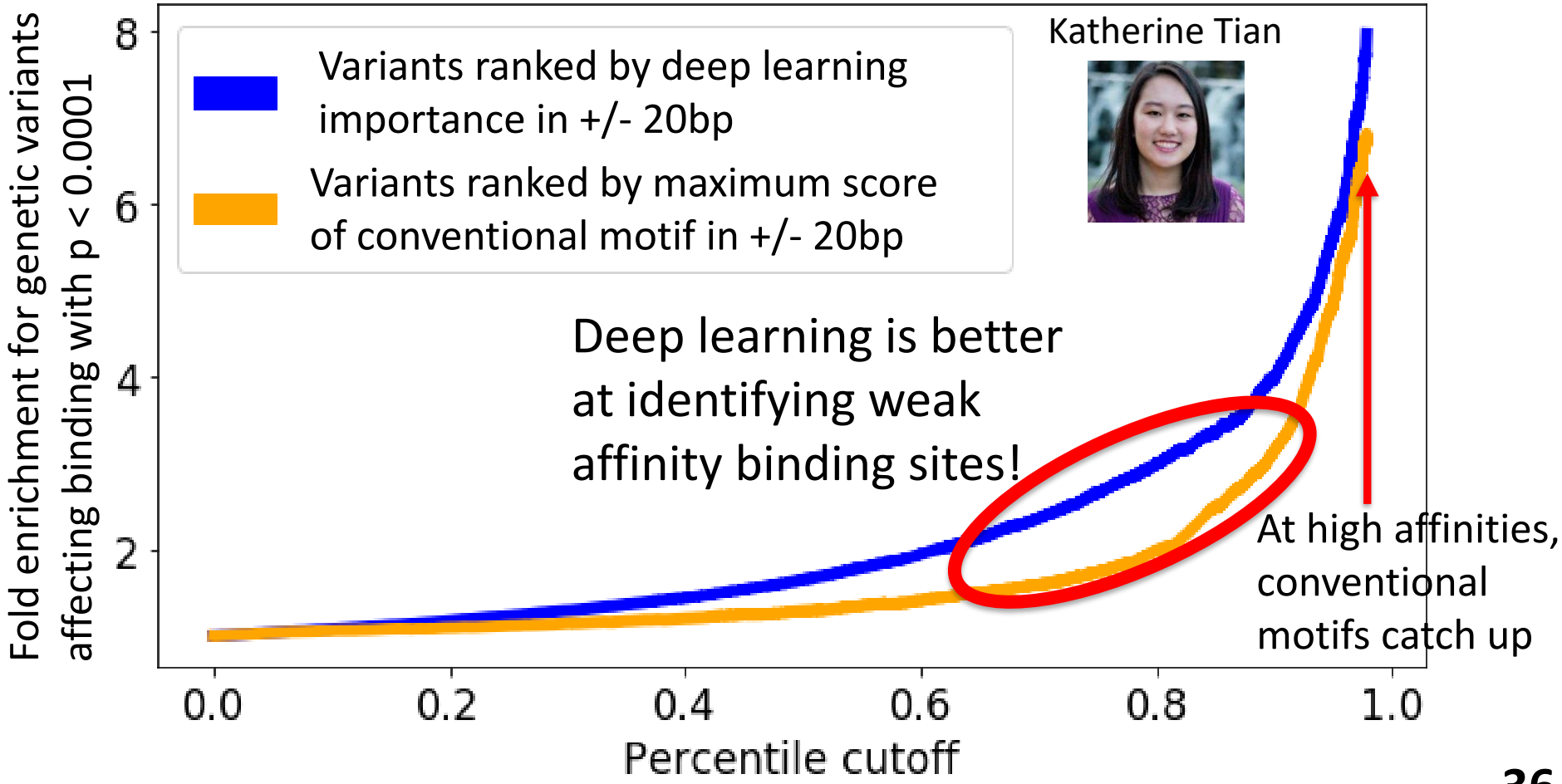


Analysis by
Abhimanyu Banerjee

Can we use interpretable deep learning to get better models of TF binding?

Revisiting our genetic variant…




Fold enrichment for genetic variants affecting binding with p < 0.0001
8
6
4
2
Variants ranked by deep learning importance in +/- 20bp
Variants ranked by maximum score of conventional motif in +/- 20bp
Katherine Tian
Deep learning is better at identifying weak affinity binding sites!
At high affinities, conventional motifs catch up
0.0
0.2
0.4
0.6
0.8
1.0
Percentile cutoff

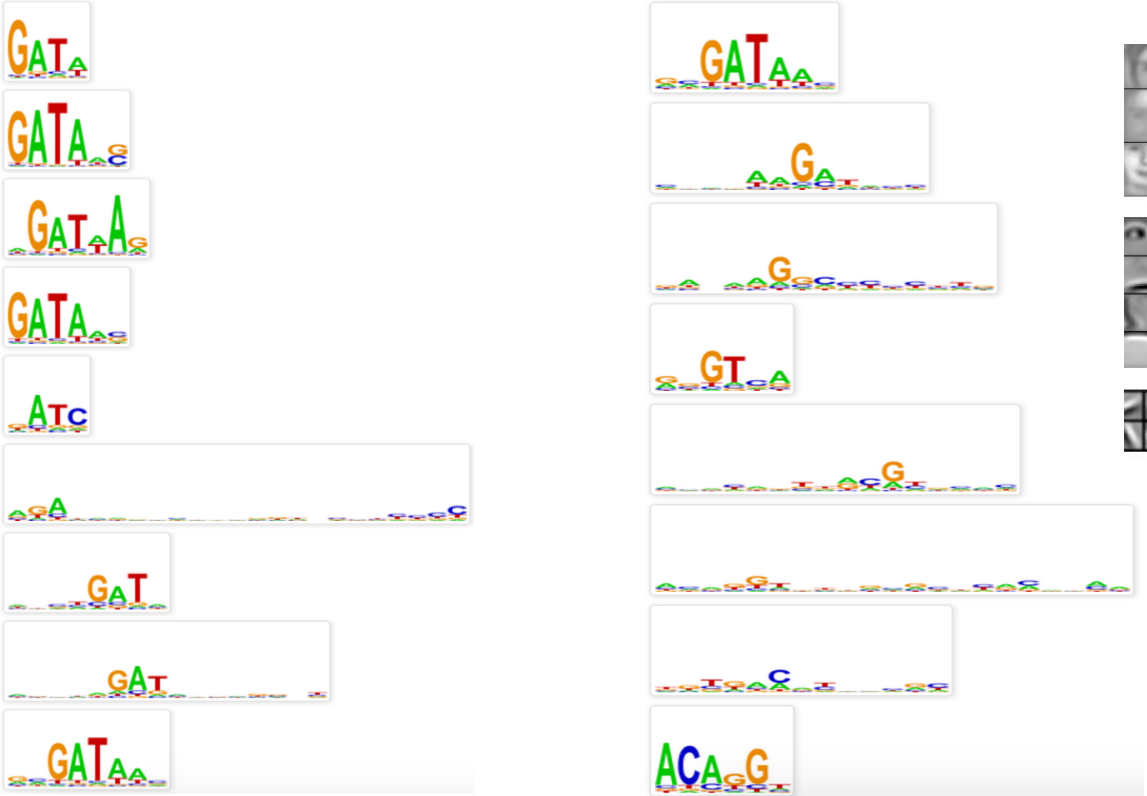
Questions for the model

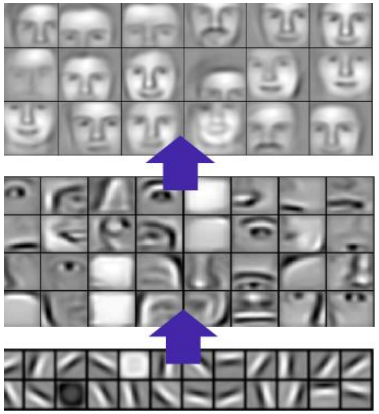
- Which parts of the input are the most important for making a given prediction?
- What are the recurring patterns in the input?

Question in biology: What are the DNA motifs driving transcription factor binding?

Naïve idea: look at individual pattern detectors

Individual GATA pattern detectors motifs found by DeepBind (Alipanahi et al.)



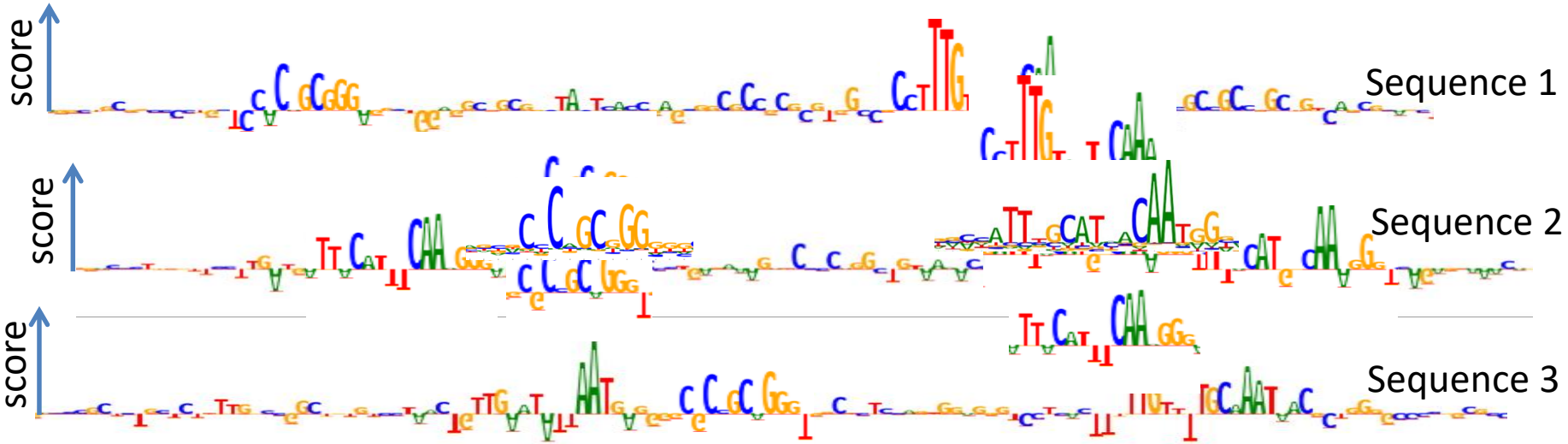


Computer vision

Problem: **High levels of redundancy, because multiple neurons cooperate with each other**

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

Insight: input-level importance scores reveal combined contributions

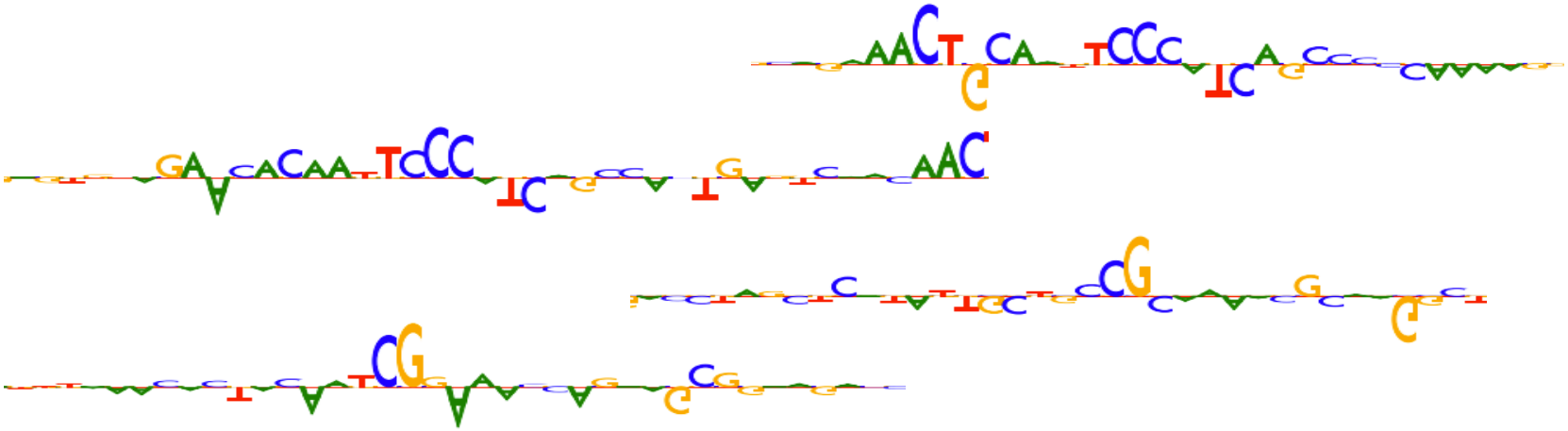


TF-MoDISco: TF **Mo**tif **D**iscovery from **I**mportance **Sco**res

https://github.com/kundajelab/tfmodisco

TF-MoDISco: More details

(1) Compute affinities between pairs of seqlets using cross-correlation-**like** metric



(2) Cluster affinity matrix

(3) Aggregate seqlets in a cluster to get motifs

Key idea: Density-Adaptive Distance (1)

Problem: notion of “far away” varies with the cluster

- Weak motif clusters: seqlets may be farther away on average
- Notion of “far” needs to take this into account

Key idea: Density-Adaptive Distance (2)

- Soln: Adapt notion of distance to the local density of the data!

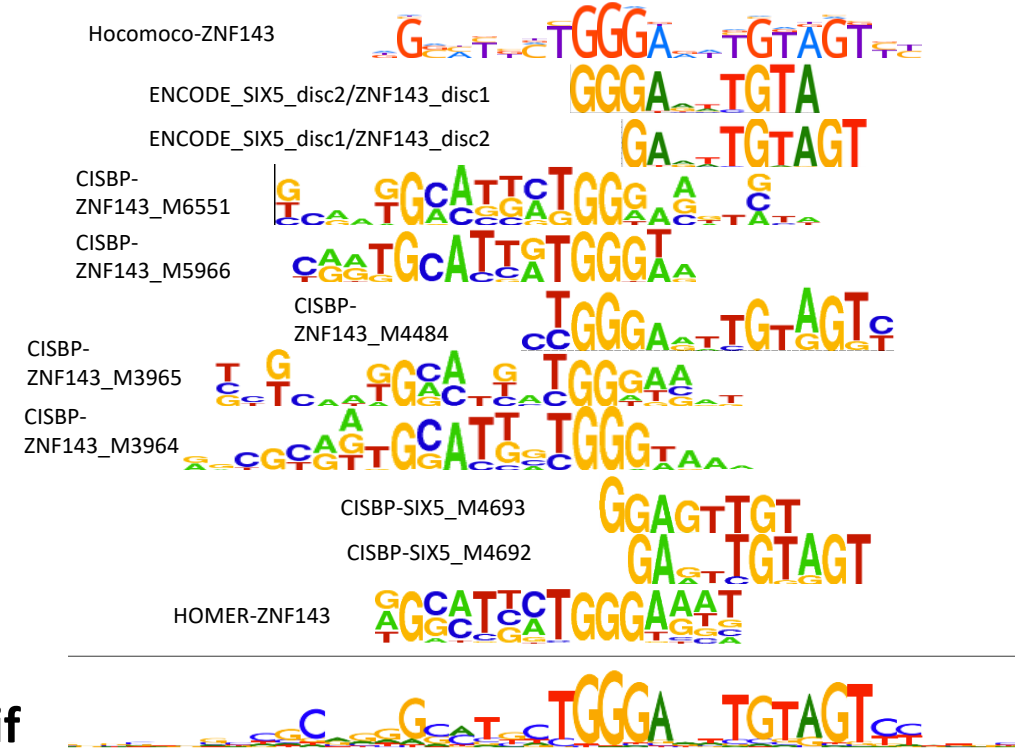
  - First step of t-sne: compute conditional probs

$$p_{j|i} \propto \exp(-\beta_i \times \text{distance}(i, j))$$

  - β_i is tuned to attain a desired perplexity!
    - Larger β_i will be used in denser region of the space
  - Supply density-adapted probabilities to multiple rounds of Louvain community detection

TF-MoDISco motifs are broader and more consolidated than traditional motifs

Known motifs for SIX5/ZNF143



Corresponding TF-MoDISco motif

Žiga
Avsec

10 bp periodic Nanog motif

Base frequency (PWM)

TF-MODISCO motif

10 bp

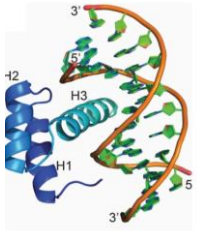
Klf4 | Nanog | Oct4 | Sox2

Avg. counts

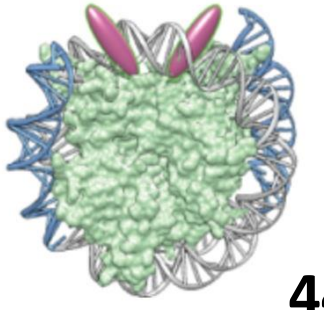
Position

Experimental evidence:

Nanog homeodomain
Hayakshi et al.
PNAS 2015



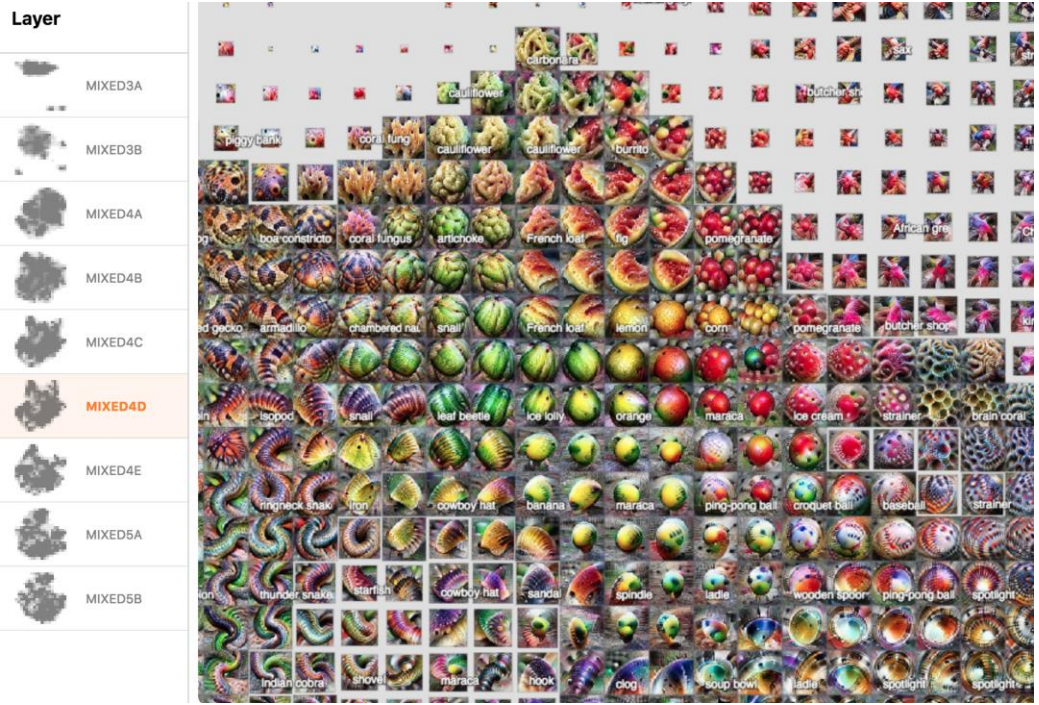
10 bp periodic binding of homeobox TFs to nucleosome DNA from recent *in vitro* NCAP-SELEX data (Zhu et al. Nature 2018)



Summary

- DeepLIFT: can efficiently reveal important parts of the input for a given prediction
  - https://github.com/kundajelab/deeplift
- TF-MoDISco: Motif Discovery from Importance Scores
  - Reveals recurring patterns in the input
  - https://github.com/kundajelab/tfmodisco
- Can be used to gain novel insights on the regulatory code of the genome

Recent work on “Activation Atlases” (OpenAI)



- https://distill.pub/2019/activation-atlas/
  - Sample vectors of filter activations on real data
  - Dimensionality reduce with t-sne; implicitly identifies filters that fire together
  - At each region of the dimensionality-reduced map, derive a visualization corresponding to the vector of filter activations present there
  - Key Drawbacks:
    - Dimensionality reduction to 2d might be missing a lot of information
    - **Does not provide clusters**

Dimensionality reduction vs clustering

One might think that a true clustering algorithm, like k-means, might produce a more robust decomposition of the activation manifold. After all, centroids produced by such an algorithm are by definition close to clusters of activations the network produces, a property that discretizing the dimensionality-reduced activations doesn't guarantee. We experimented with different clustering techniques such as k-means, spherical k-means, and DBSCAN. However, the images produced by visualizing the resulting centroids were subjectively worse and less interpretable than the technique described in this article. Also, dimensionality reduction followed by 2D binning allows for multiple levels of detail while preserving spatial consistency, which is necessary for making atlasses zoomable. Thus, we preferred that method over clustering in this article — but finding tradeoffs between these techniques remains an open question.

- I too found that t-sne was able to separate clusters better than k-means, DBSCAN, spectral clustering, etc…
- Plugging t-sne's trick of density adaptation into Louvain successfully recapitulated the structure of t-sne.

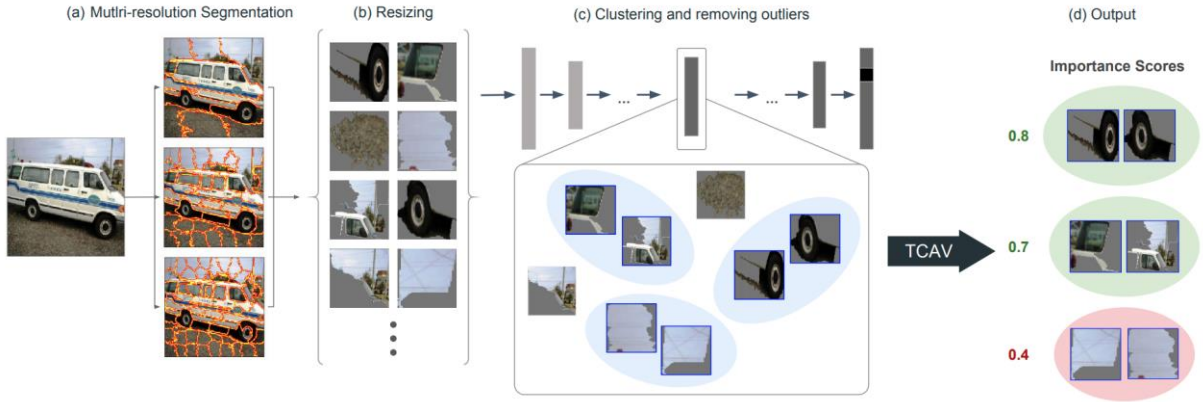
Recent work on discovering "concept activation vectors" (Google Brain)

Automating interpretability: discovering and testing visual concepts learned by neural networks

Amirata Ghorbani*
Stanford University
amiratag@stanford.edu

James Wexler
Google Brain
jwexler@google.com

Been Kim
Google Brain
beenkim@google.com



- Approach
  - Segment image
  - Resize segments to fill entire input, feed through network
  - Cluster segments based on activation of bottleneck layer
- Drawbacks
  - Classifier must give reasonable results when patch is resized to fill image
  - Crude clustering: *"The best results…were acquired using k-means clustering followed by removing all points but the n points that have the smallest L2 distance from the cluster center"*

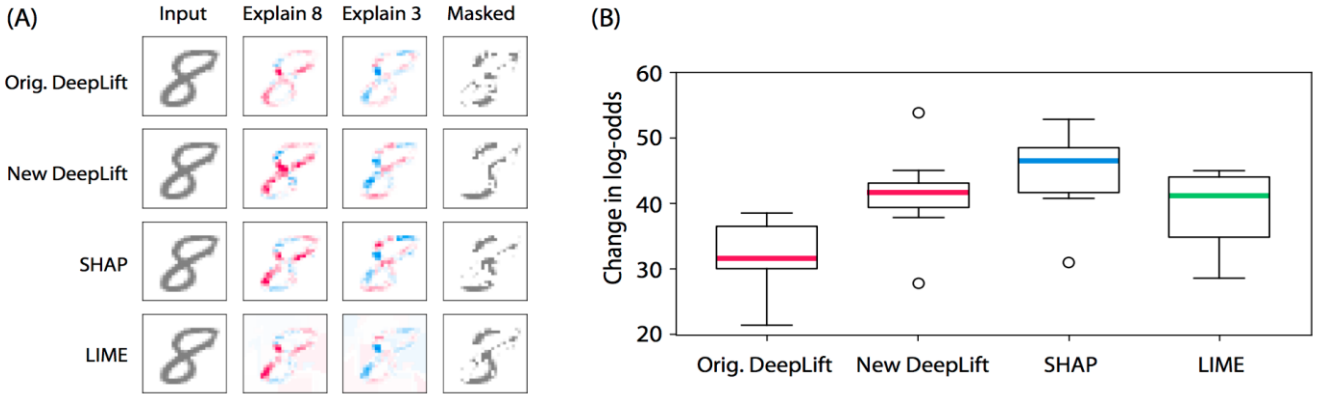
Shapely values

- Comes from game theory; Shapely values assign contributions to players in cooperative games.
  - Look at all possible orderings of including players in the game
  - For each ordering, find marginal change in reward when a player is included
  - Average a player's marginal contribution to reward over all orderings
- Analogy for model importance:
  - "reward" is model output
  - "players" are individual inputs
  - "including" an input means setting it to its actual value vs. sampling it from some background distribution

SHAP values: more efficient Shapely approx.

- SHAP values (Lundberg & Lee, NIPS 2017) proposed more efficient way to estimate Shapely contributions by performing weighted linear regression.
- Still requires a large number of samples to provide decent results!
- In paper, to interpret a single MNIST digit, used **50,000 model evaluations**

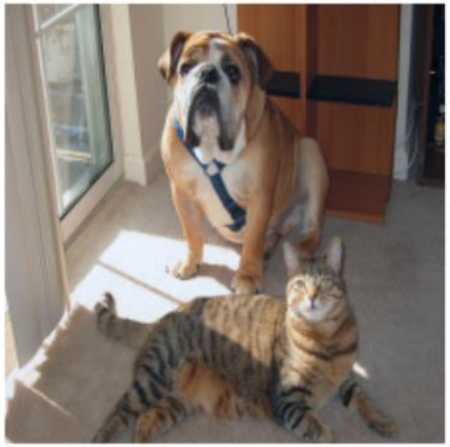
(New DeepLIFT = RevealCancel rule)



- For efficiency, proposed a hybrid of SHAP and DeepLIFT called DeepSHAP
  - Handles some operations that DeepLIFT doesn't handle (e.g. elementwise multiplications). Current implementation doesn't have RevealCancel rule. Reduces to DeepLIFT without RevealCancel rule for many standard architectures.

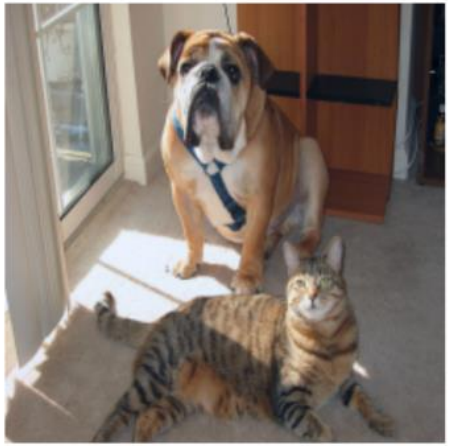
Tip: Beware GuidedBackprop and DeconvNet!

- These backprop-based methods do not produce class-specific visualizations (theoretically proven)



(a) Original Image

(b) Guided Backprop 'Cat'

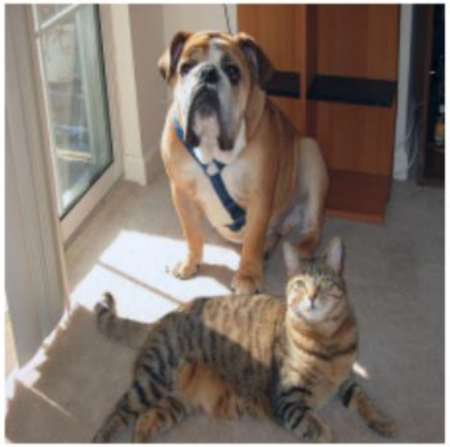


(g) Original Image

(h) Guided Backprop 'Dog'

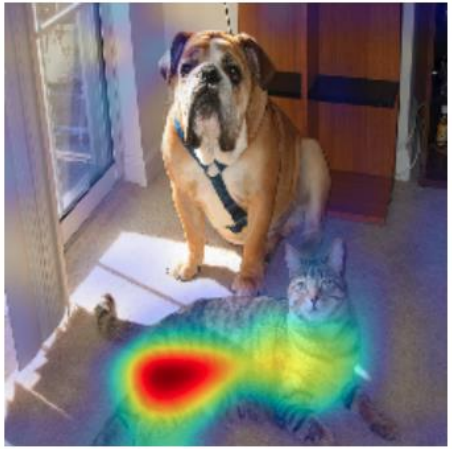
Tip: Beware GuidedBackprop and DeconvNet!

- These backprop-based methods do not produce class-specific visualizations (theoretically proven)
- Is possible to introduce class-specificity to GuidedBackprop through multiplying with “class activation maps” (CAM)
  - Idea of CAM: for some higher-level convolutional layer, assign class-specific importance to each channel (“feature map”) using gradients

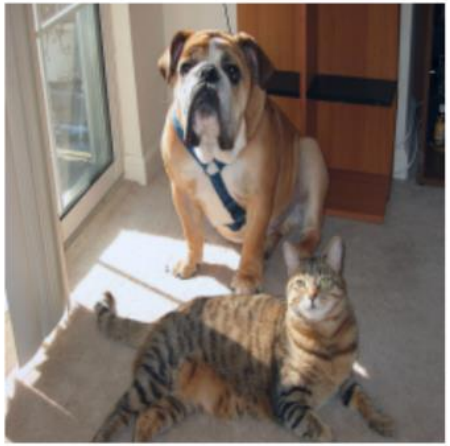


(a) Original Image

(b) Guided Backprop 'Cat'



(c) Grad-CAM 'Cat'



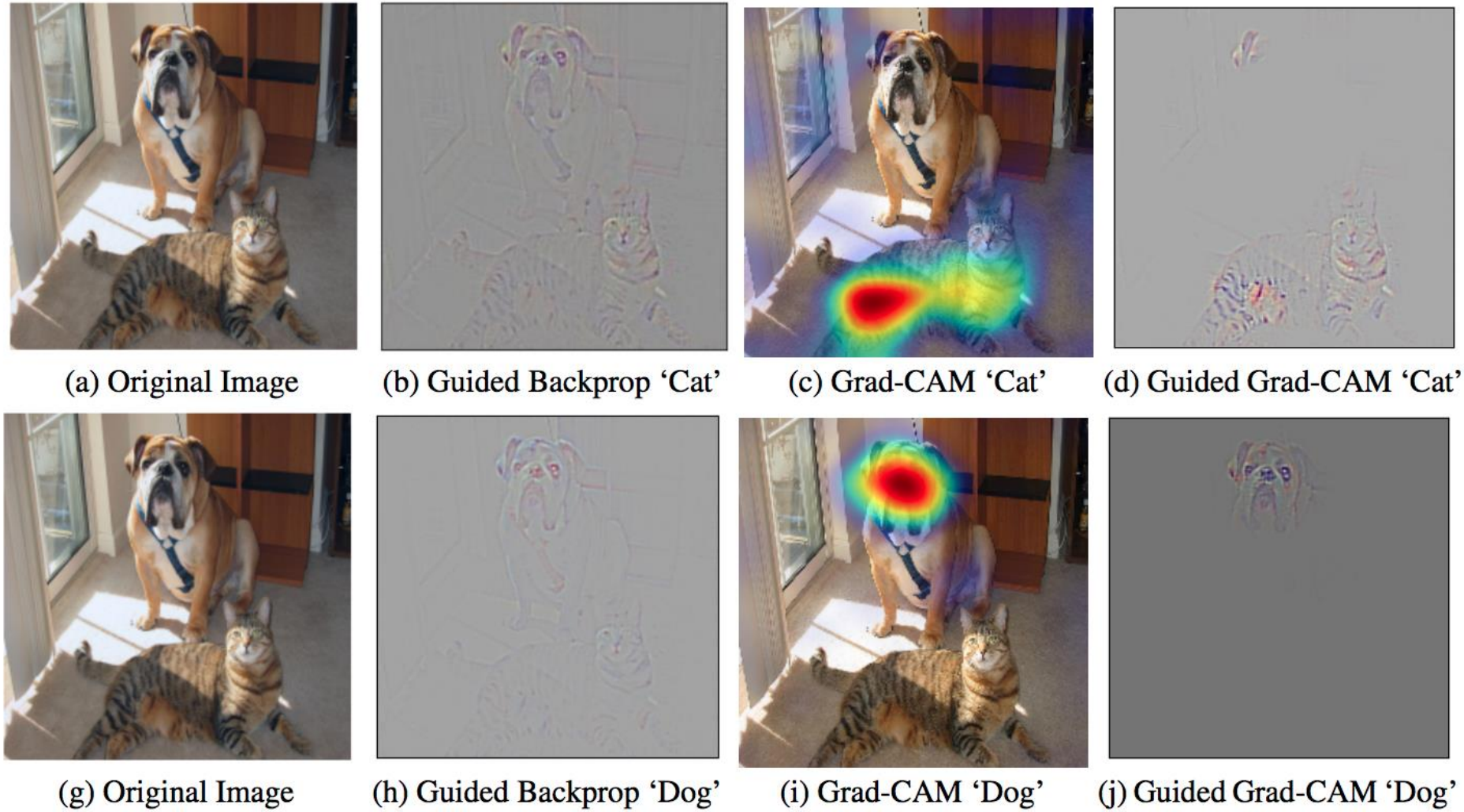
(g) Original Image

(h) Guided Backprop 'Dog'

(i) Grad-CAM 'Dog'

Tip: Beware GuidedBackprop and DeconvNet!

- These backprop-based methods do not produce class-specific visualizations (theoretically proven)
- Is possible to introduce class-specificity to GuidedBackprop through multiplying with "class activation maps" (CAM)
  - Idea of CAM: for some higher-level convolutional layer, assign class-specific importance to each channel ("feature map") using gradients
  - Do elementwise multiplication with GuidedBackprop to introduce class-specificity
  - Method is called "Guided Grad-CAM"



(a) Original Image (b) Guided Backprop 'Cat' (c) Grad-CAM 'Cat' (d) Guided Grad-CAM 'Cat'

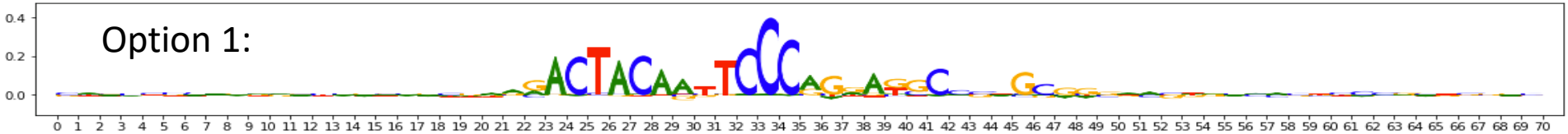
(g) Original Image (h) Guided Backprop 'Dog' (i) Grad-CAM 'Dog' (j) Guided Grad-CAM 'Dog'

Key idea 1: Correlation alternative

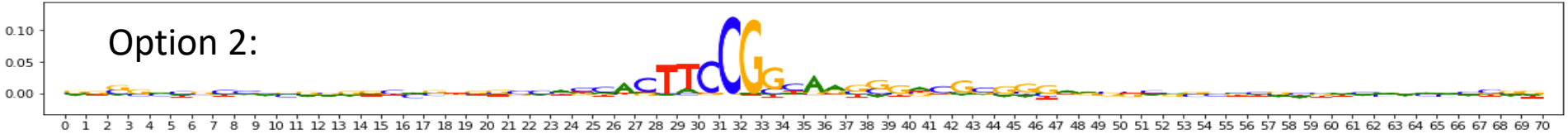
input:

Which pattern is the input a better match to?

Option 1:

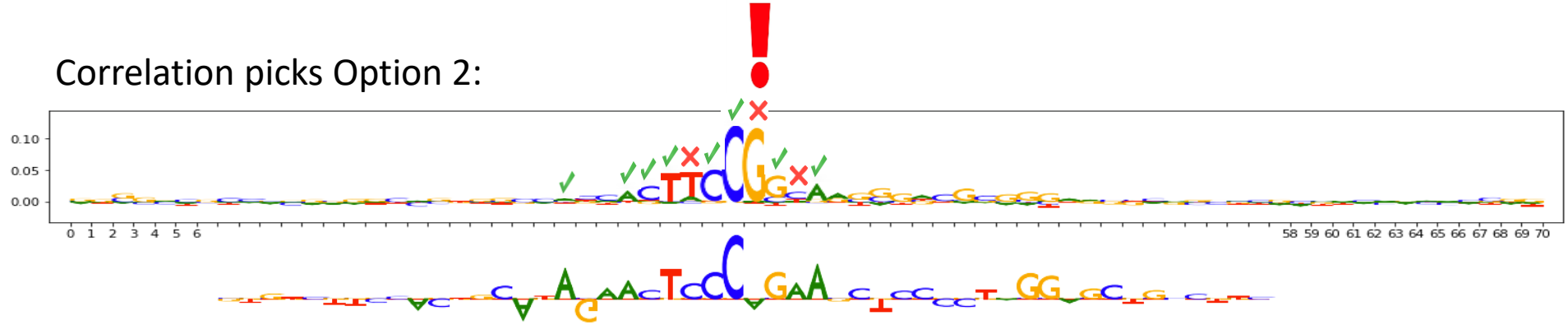


Option 2:

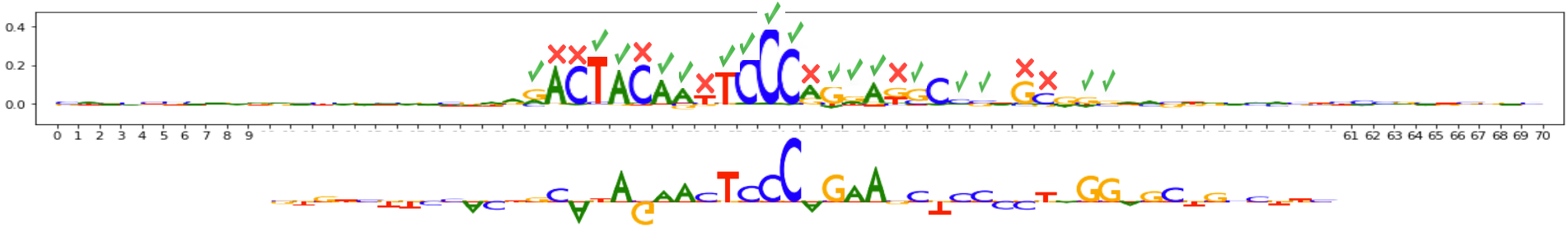


Key idea 1: Correlation alternative

Correlation picks Option 2:



Our metric ("Continuous Jaccard") picks Option 1:



Key idea 1: Correlation alternative

- **What is the issue with correlation?**
  - Correlation involves element-wise products:

$$\frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

  - Polynomial degree 2: agreement at a few largest-magnitude positions preferred to agreement at several smaller-magnitude positions
  - Input = **(-1, -1, -2, 4, -1, -1, -1)**
    - Correlation with **(0, 0, 0, 4, 0, 0, 0)** = **0.98**
    - Correlation with **(-1, -1, -2, 0, -1, -1, -1)** = **0.87**

Key idea 1: Cross-correlation alternative

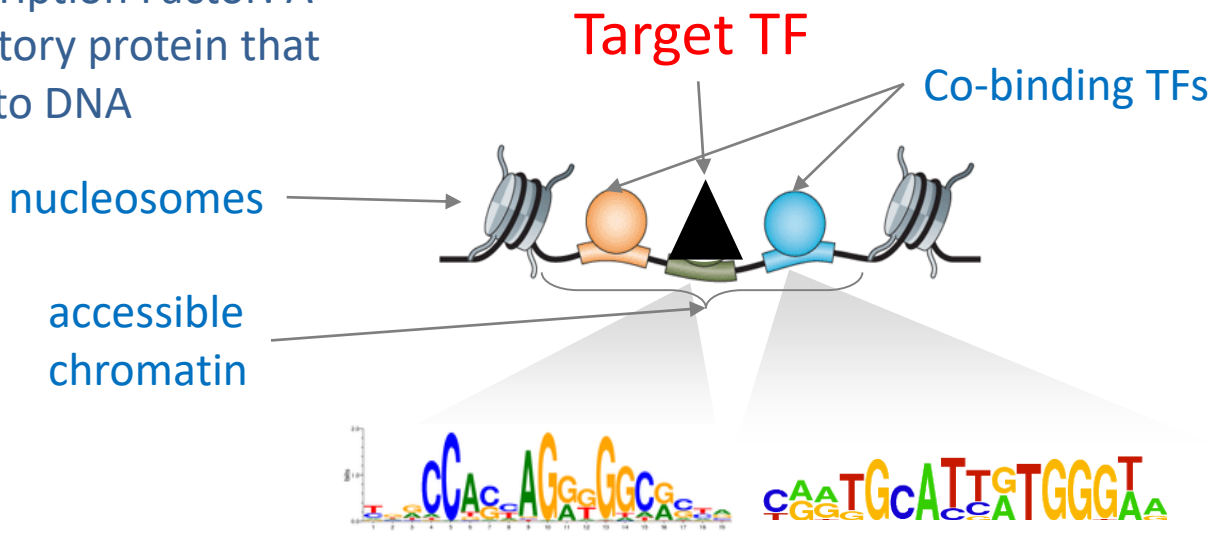
- **Continuous Jaccard: like Jaccard distance for reals**

$$x \cap y = \min(|x|, |y|) \times \text{sign}(x) \times \text{sign}(y)$$
$$x \cup y = \max(|x|, |y|)$$

  - "Continuous Jaccard" = $\dfrac{\sum_i x_i \cap y_i}{\sum_i x_i \cup y_i}$
  - Input = **(-1, -1, -2, 4, -1, -1, -1)**
    - Contin. Jaccard with **(0, 0, 0, 4, 0, 0, 0)** = **4/11**
    - Contin. Jaccard with **(-1, -1, -2, 0, -1, -1, -1)** = **7/11**

Goal: Understand the DNA patterns ("motifs") determining *in vivo* transcription factor binding

Transcription Factor: A regulatory protein that binds to DNA



learn predictive sequence motifs

Adapted from Shlyueva et al. (2014) Nature Reviews Genetics.

Backup