

Interpreting noncoding variants

BMI/CS 776

www.biostat.wisc.edu/bmi776/

Spring 2019

Colin Dewey

colin.dewey@wisc.edu

Goals for lecture

Key concepts

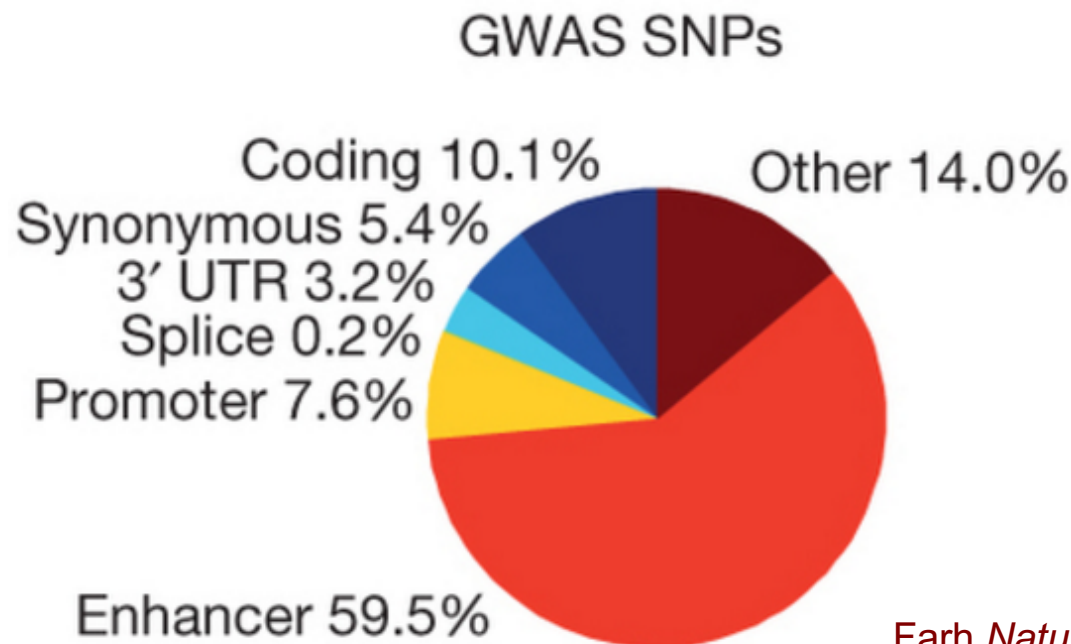
- Mechanisms disrupted by noncoding variants
- Deep learning to predict epigenetic impact of noncoding variants

GWAS output

- GWAS provides list of SNPs associated with phenotype
- SNP in coding region
 - Link between the protein and the disease?
- SNP in noncoding region
 - What genes are affected?

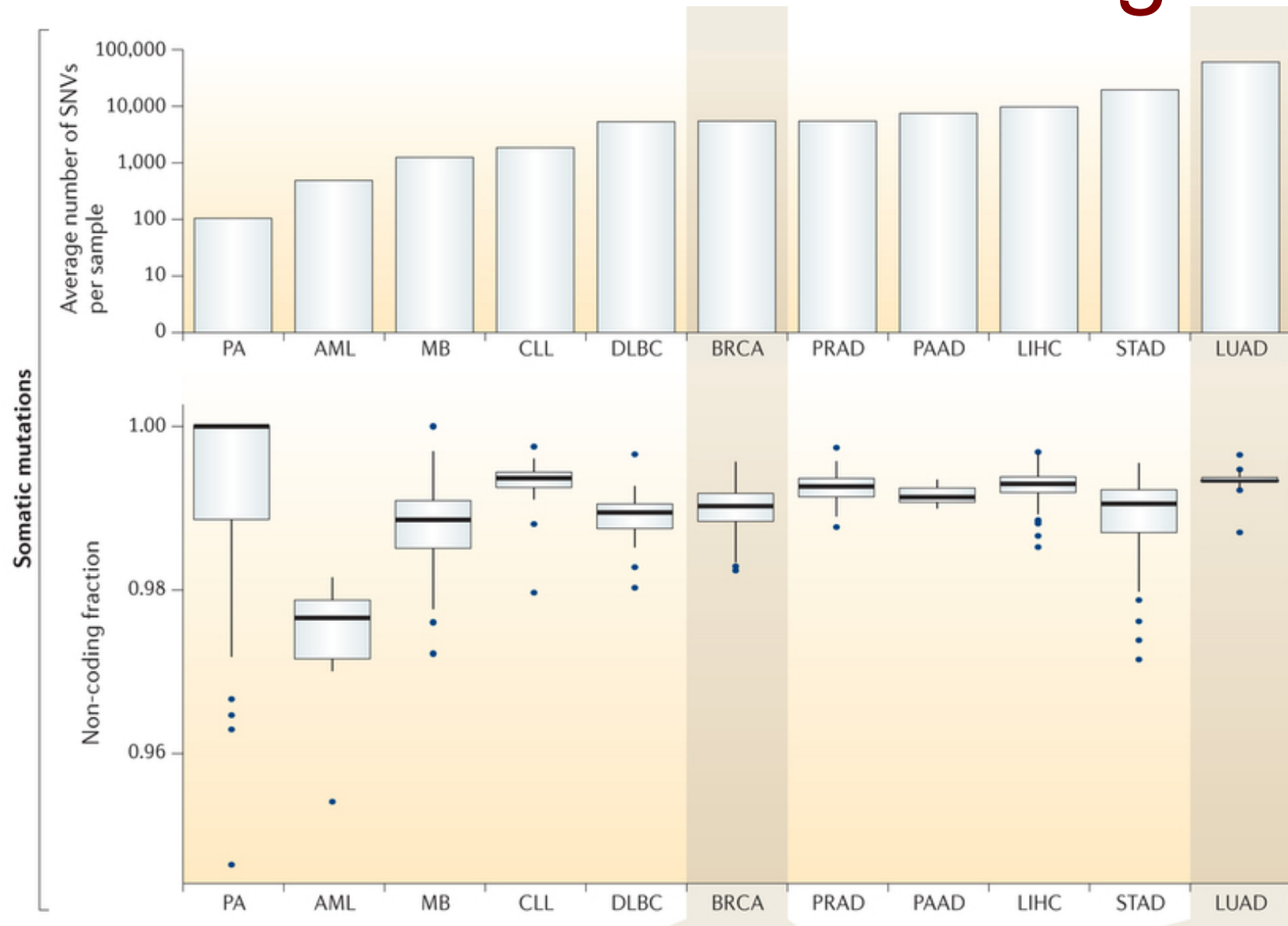
Noncoding variants common in GWAS

- Meta-analysis of GWAS for 21 autoimmune diseases
 - Rheumatoid arthritis, lupus, multiple sclerosis, etc.
- Method to prioritize candidate causal SNPS
- **90%** of causal variants are noncoding



Farh *Nature* 2015

Almost all single nucleotide variants in cancer are noncoding



Khurana
*Nature
Reviews
Genetics
2016*

However, very few of these are driver mutations

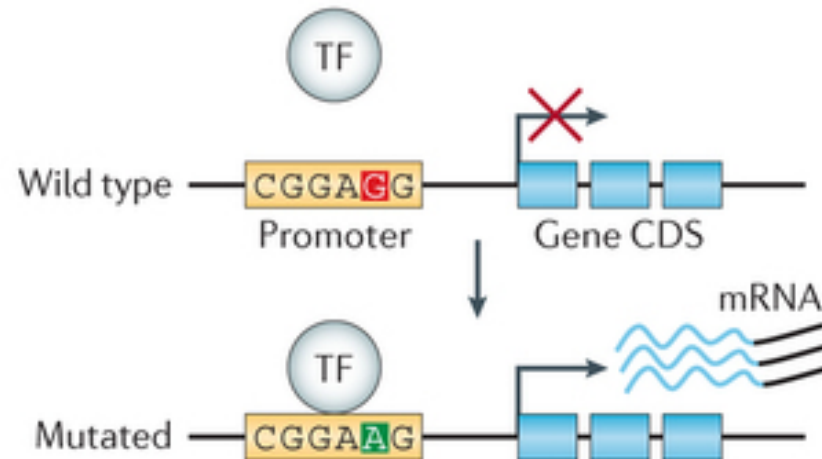
Ways a noncoding variant can be functional

- Disrupt DNA sequence motifs
 - Promoters, enhancers
- Disrupt miRNA binding
- Mutations in introns affect splicing
- Indirect effects from the above changes

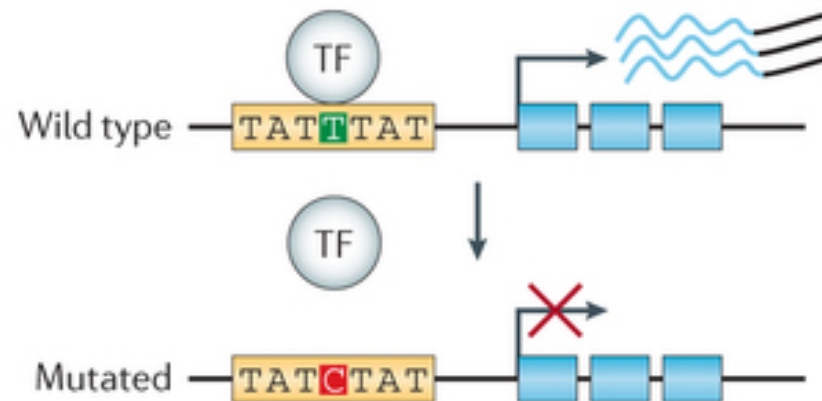
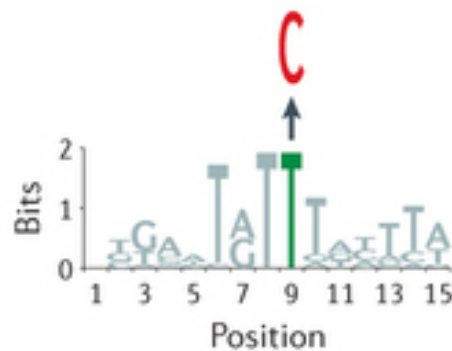
Examples in Ward and Kellis *Nature Biotechnology* 2012

Variants altering motifs

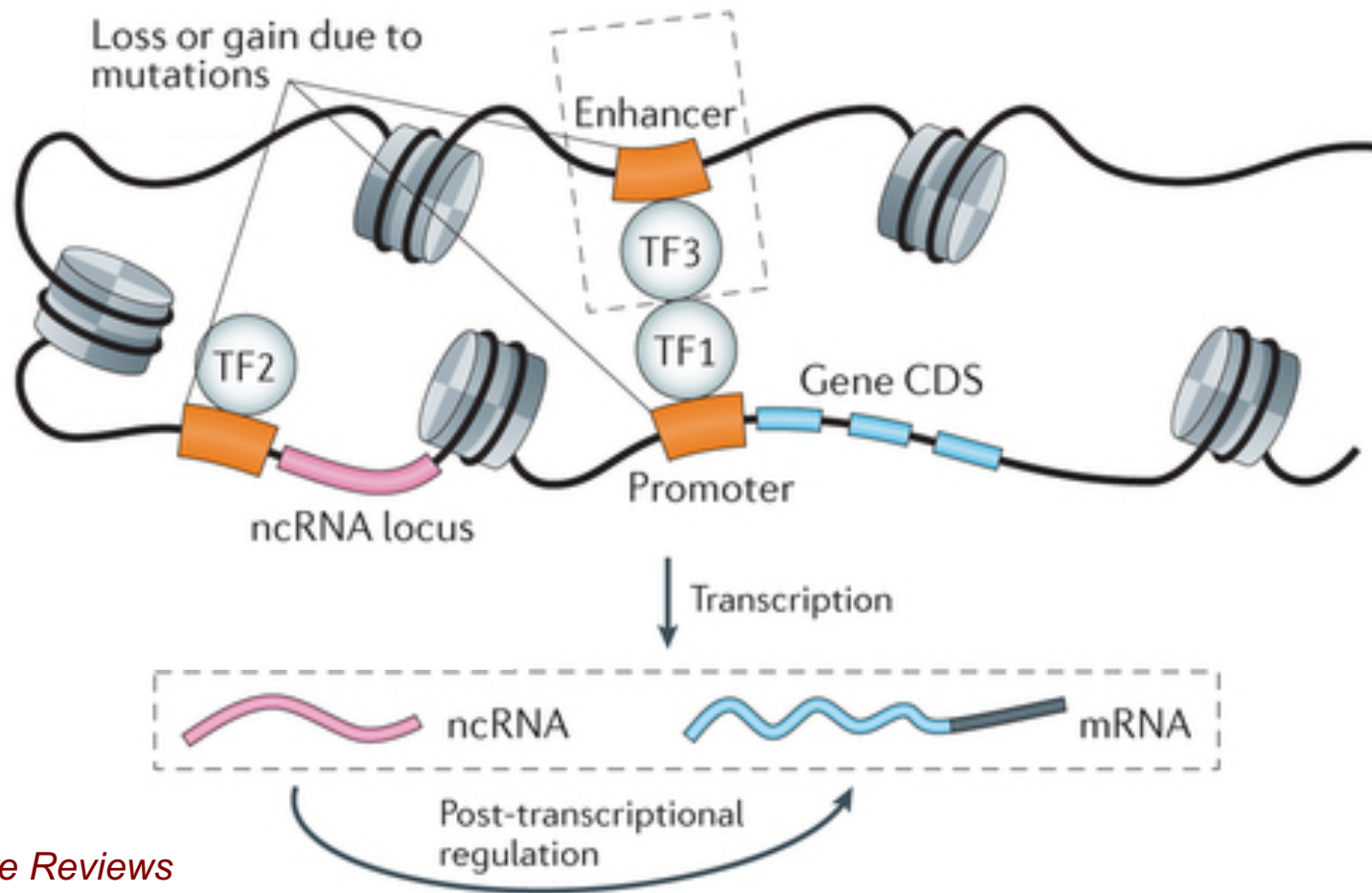
Gain of motif



Loss of motif



Variants affect proximal and distal regulators

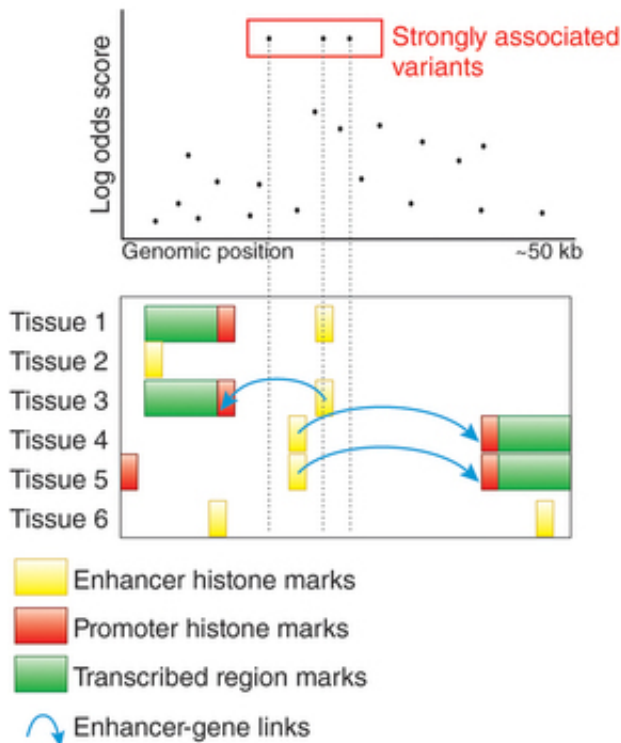


Khurana *Nature Reviews*
Genetics 2016

Evidence used to prioritize noncoding variants

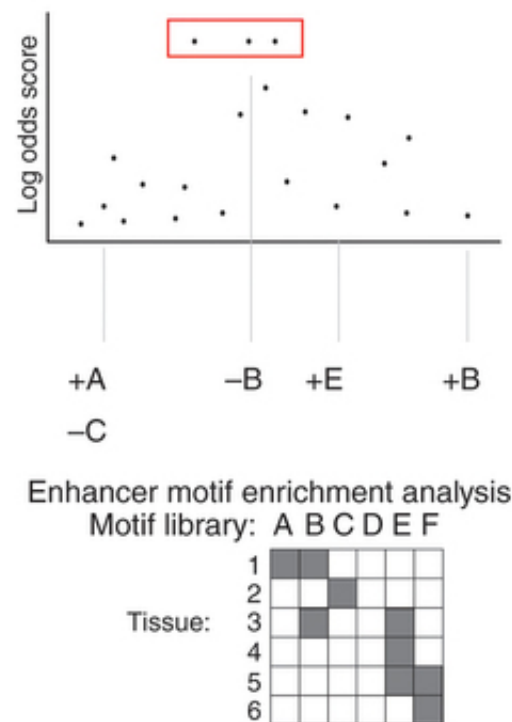
Interpreting GWAS signals using functional and comparative genomics datasets

a Dissect associated haplotype using functional genomics



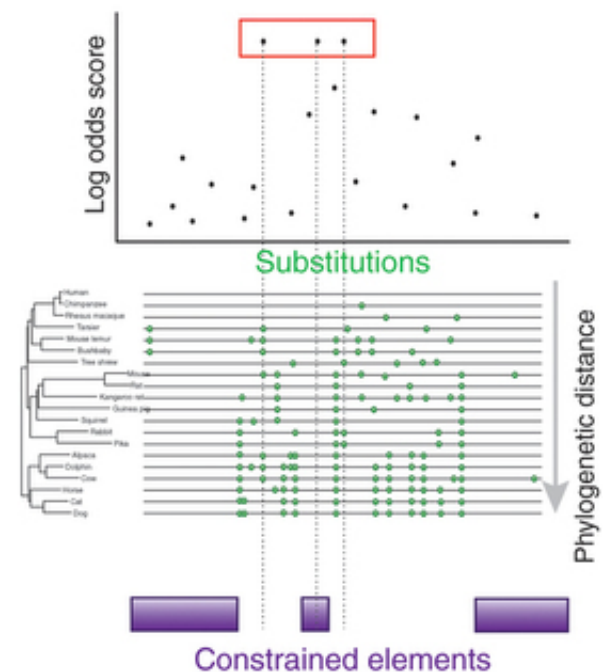
Chromatin state annotations

b Dissect associated haplotype using regulatory genomics



Motifs altered by variants

c Dissect associated haplotype using comparative genomics



Mammalian constraint

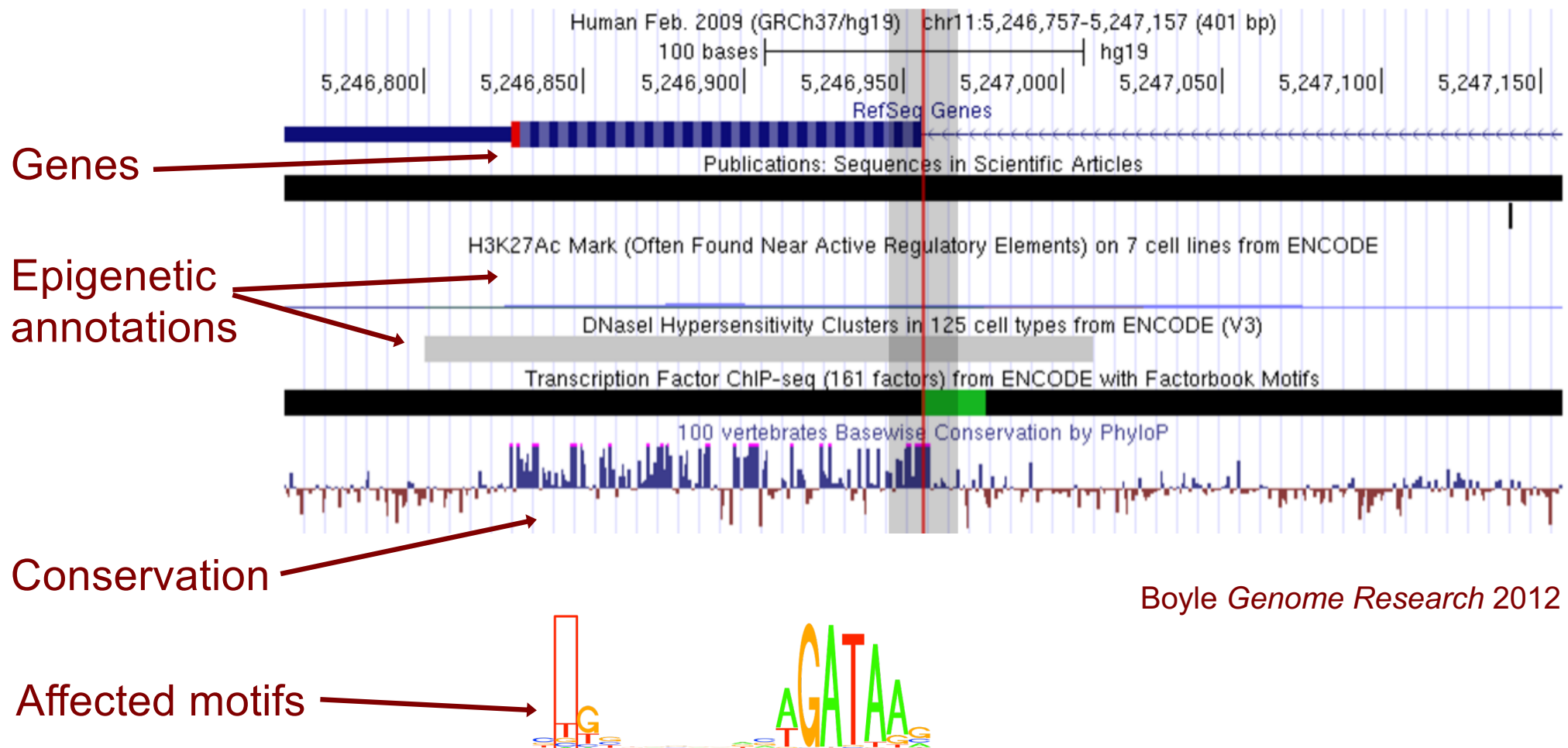
Visualizing evidence



Data supporting chr11:5246957 (rs33914668)

Summary of evidence → **Score: 2a**

Likely to affect binding



Boyle Genome Research 2012

Combined Annotation– Dependent Depletion (CADD)

- Example of an algorithm that integrates multiple types of evidence into a single score
 - Conservation
 - Epigenetic information
 - Protein function scores for coding variants
- Train support vector machine on simulated and observed variants
- Variants present in simulation but not observed are likely deleterious

Prioritizing variants with epigenetics summary

- + Disrupted regulatory elements one of the best understood effects of noncoding SNPs
- + Make use of extensive epigenetic datasets
- + Similar strategies have actually worked
 - rs1421085 in *FTO* region and obesity
 - Claussnitzer *New England Journal of Medicine* 2015
- Epigenetic data at a genomic position is often in the presence of the reference allele
 - Don't have measurements for the SNP allele

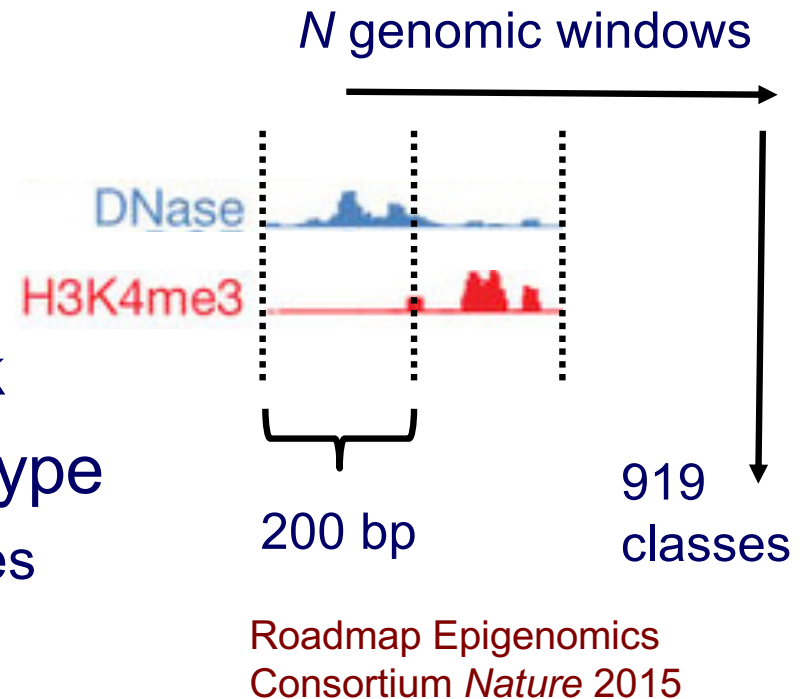
DeepSEA

- Given:
 - A sequence variant and surrounding sequence context
- Do:
 - Predict TF binding, DNase hypersensitivity, and histone modifications in multiple cell and tissue types
 - Predict variant functionality

Zhou and Troyanskaya *Nature Methods* 2015

Classifier input and output

- Output
 - 200 bp windows of genome
 - Label 1 if window contains peak
 - Label for each epigenetic data type
 - Multiple types of epigenetic features
 - Multiple types of cells and tissues



- Input: 1000 bp DNA sequence centered at window

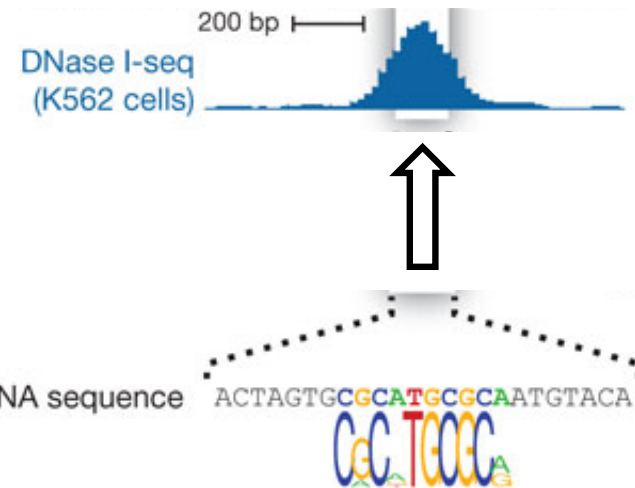
$$x_i =$$

index	1	...	401	402	403	...	1000
A	0		1	0	0		0
C	0		0	0	0		1
G	1		0	1	1		0
T	0		0	0	0		0

Desired properties for epigenomic classifier

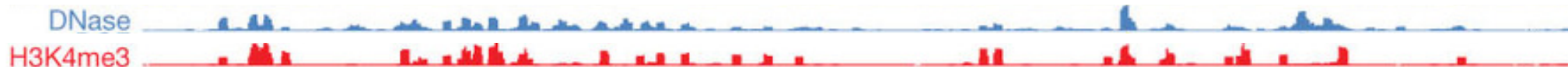
- Learn preferences of DNA-binding proteins
 - Locally: “motifs” and other simple sequence patterns
 - Sequence context: “*cis*-regulatory modules”

- Support nonlinear decision boundaries



Neph *Nature* 2012

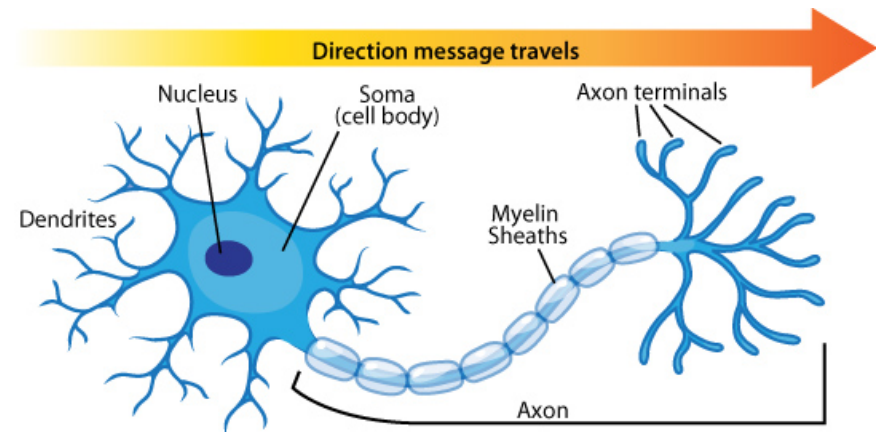
- Multiple, related prediction tasks



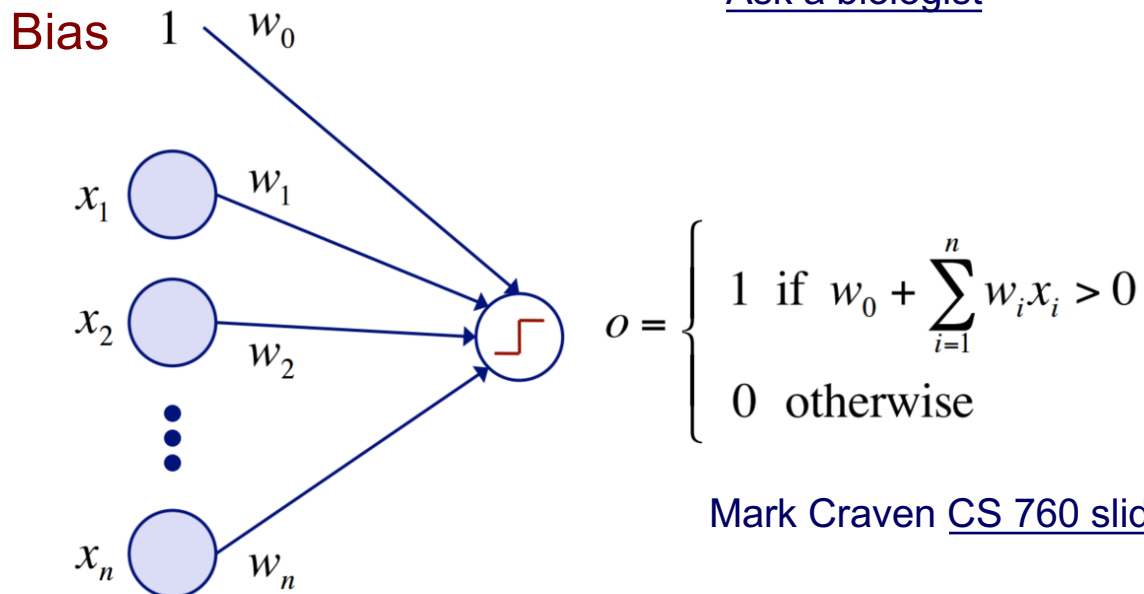
Roadmap Epigenomics Consortium *Nature* 2015

Perceptron

- Inspired by neuron
- Simple binary classifier
 - Linear decision boundary



Ask a biologist



Mark Craven CS 760 slides

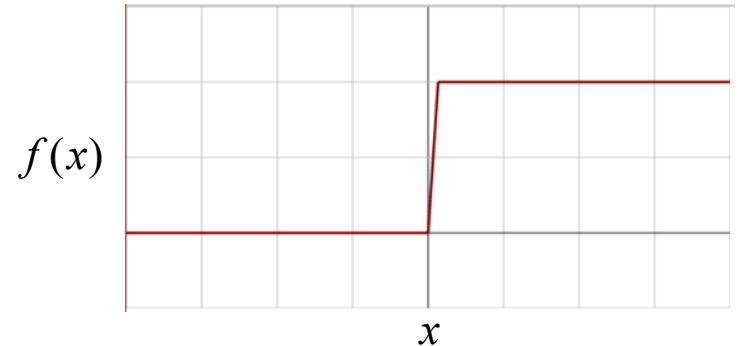
$$x_{1,A} \dots x_{1000,T}$$

Activation function

- What makes the neuron “fire”?

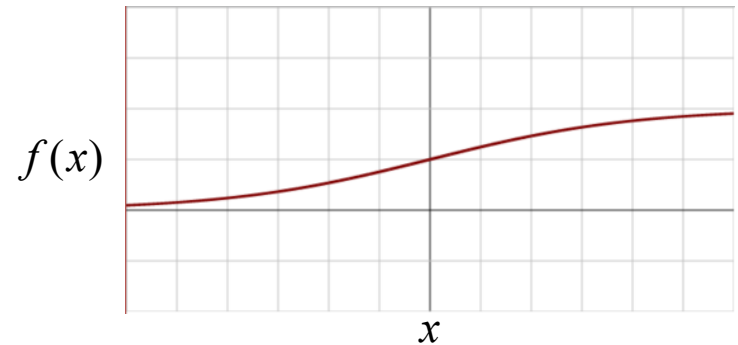
- Step function

$$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$$



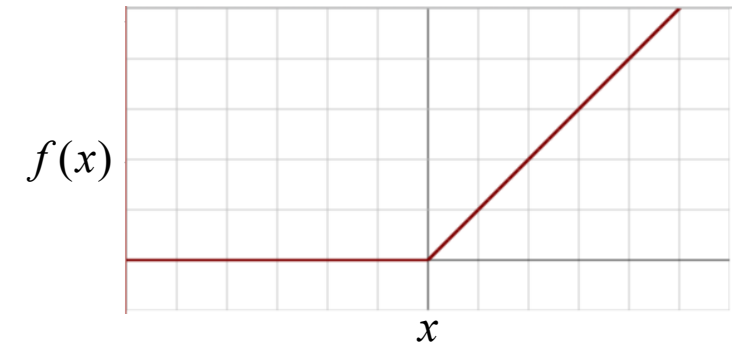
- Sigmoid function

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



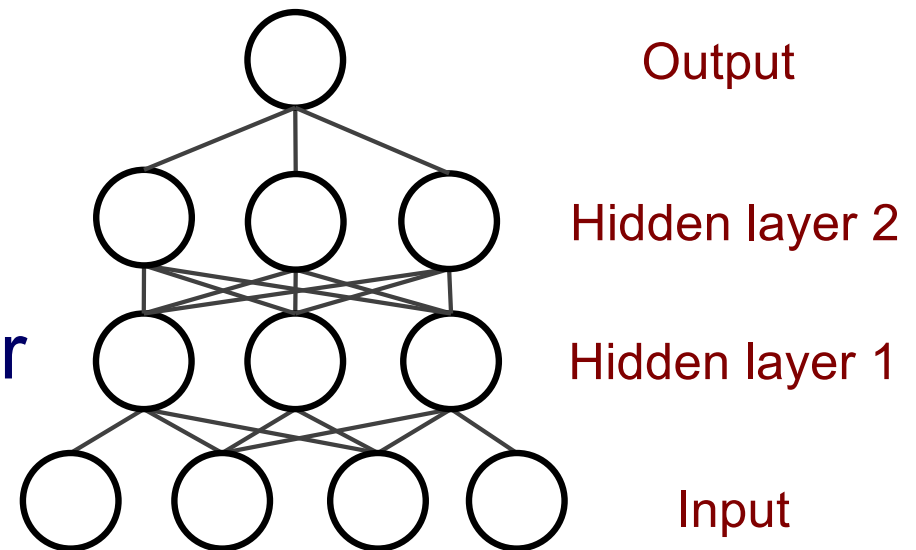
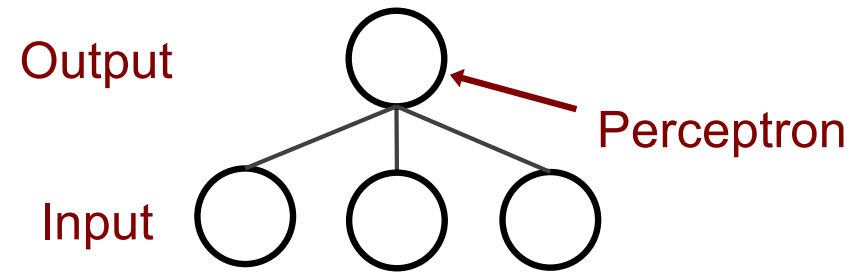
- Rectified linear unit (ReLU)

$$f(x) = \max(0, x)$$



Neural networks

- Single perceptron not useful in practice
- Neural network combines layers of perceptrons
- Learn “hidden” features
- Complex decision boundary
- Tune weights to reduce error
- Train with backpropagation
 - Stanford’s [CS231n materials](#)
 - Andrej Karpathy’s [gentle introduction](#)
 - [CS 760 slides](#)

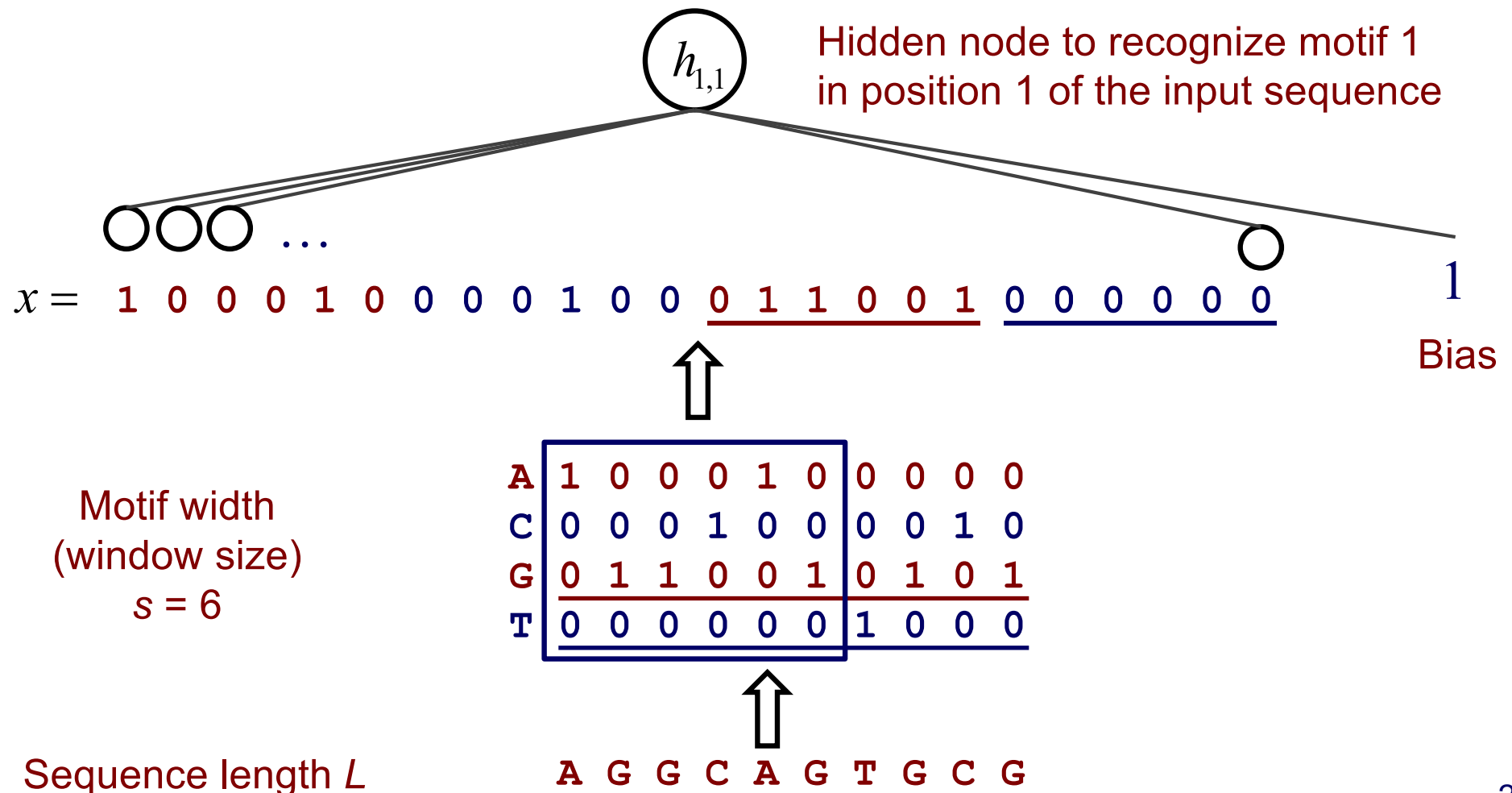


Neural network examples

- Simple linear decision boundary
- Linear decision boundary fails for XOR
- XOR with one hidden layer
- Complex, non-linear patterns
- Try varying weights, hidden units, and layers
 - What patterns can you learn with 0 hidden layers?
 - 1 hidden layer?
 - More?

First hidden layer

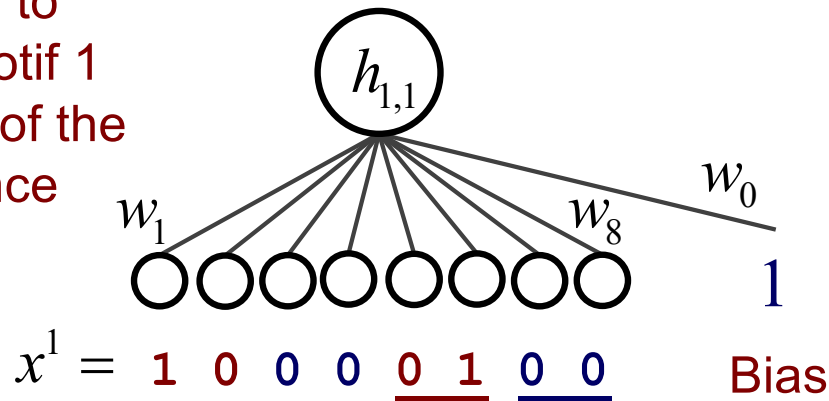
- First hidden layer scans input sequence
- Activation function fires if “motif” is recognized



First hidden layer example

- Forward pass at the first hidden layer, **input 1**

Hidden node to recognize motif 1 in position 1 of the input sequence



For simplicity,
 $s = 2$

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

**Input
sequence 1**

A G G C A G T G C G

Current weight vector for $h_{1,1}$

$$w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \end{bmatrix} = \begin{bmatrix} -1.5 \\ 2 \\ 0.1 \\ 0.4 \\ 0.2 \\ -1.7 \\ 3 \\ -0.5 \\ 0.3 \end{bmatrix}$$

Weights times input

$$w \cdot x^1 = -1.5 + 2 + 3 = 3.5$$

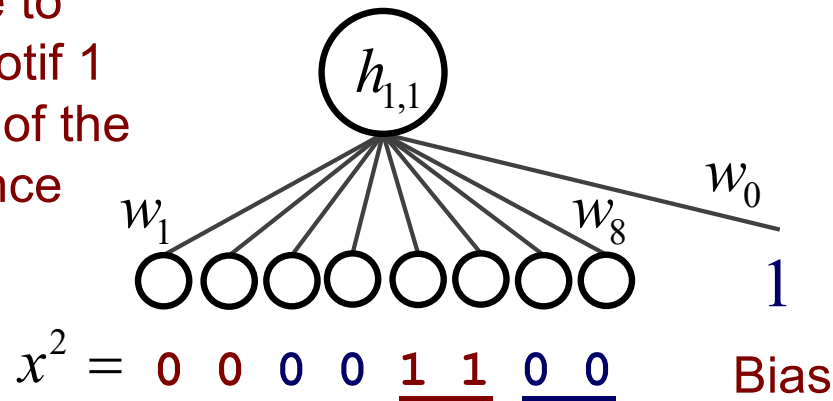
Apply ReLU activation function

$$\max(0, 3.5) = 3.5$$

First hidden layer example

- Forward pass at the first hidden layer, **input 2**

Hidden node to recognize motif 1 in position 1 of the input sequence



Input sequence 2

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

G G G A C A A C T G

Current weight vector for $h_{1,1}$

$$w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \end{bmatrix} = \begin{bmatrix} -1.5 \\ 2 \\ 0.1 \\ 0.4 \\ 0.2 \\ -1.7 \\ 3 \\ -0.5 \\ 0.3 \end{bmatrix}$$

Weights times input

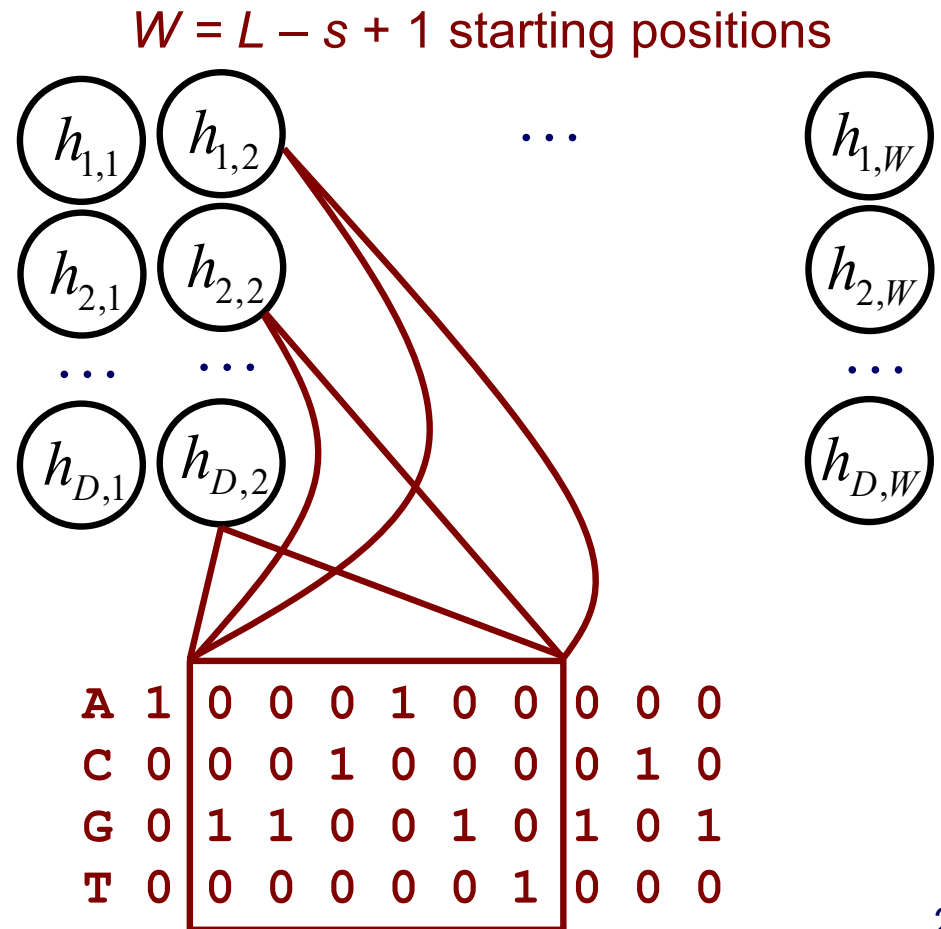
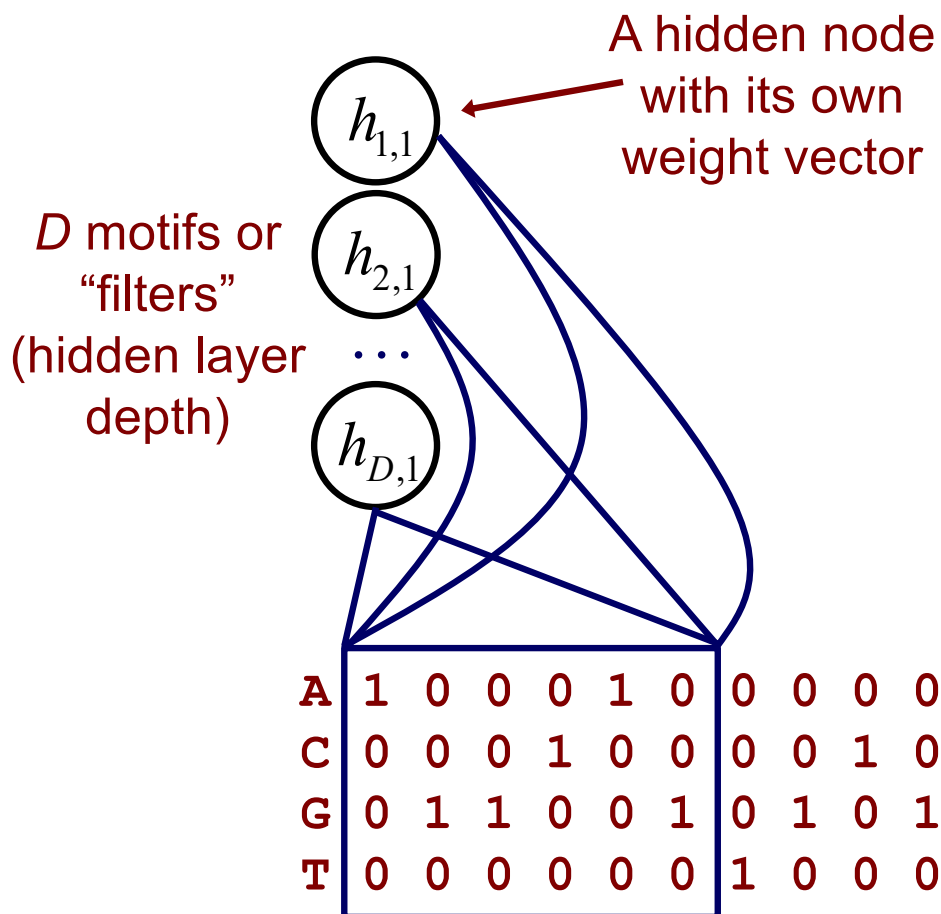
$$w \cdot x^2 = -1.5 - 1.7 + 3 = -0.2$$

Apply ReLU activation function

$$\max(0, -0.2) = 0$$

First hidden layer

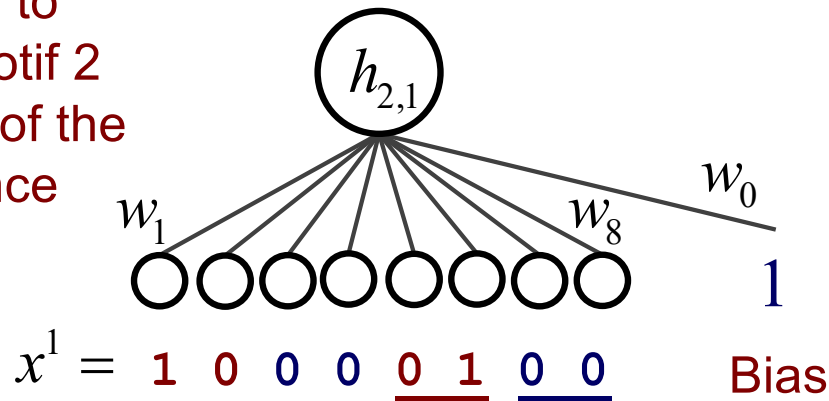
- Multiple hidden nodes to recognize different motifs at a particular position
- Check for motif at each position in sequence



First hidden layer example

- Next filter applied to **input 1**

Hidden node to recognize motif 2 in position 1 of the input sequence



Current weight vector for $h_{2,1}$

$$w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.4 \\ 2.6 \\ 0.1 \\ 0.3 \\ -0.6 \\ 0.3 \\ 5 \\ 3 \end{bmatrix}$$

Weights times input

$$w \cdot x^1 = 0.5 + 0.4 + 0.3 = 1.2$$

Apply ReLU activation function

$$\max(0, 1.2) = 1.2$$

Input sequence 1

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

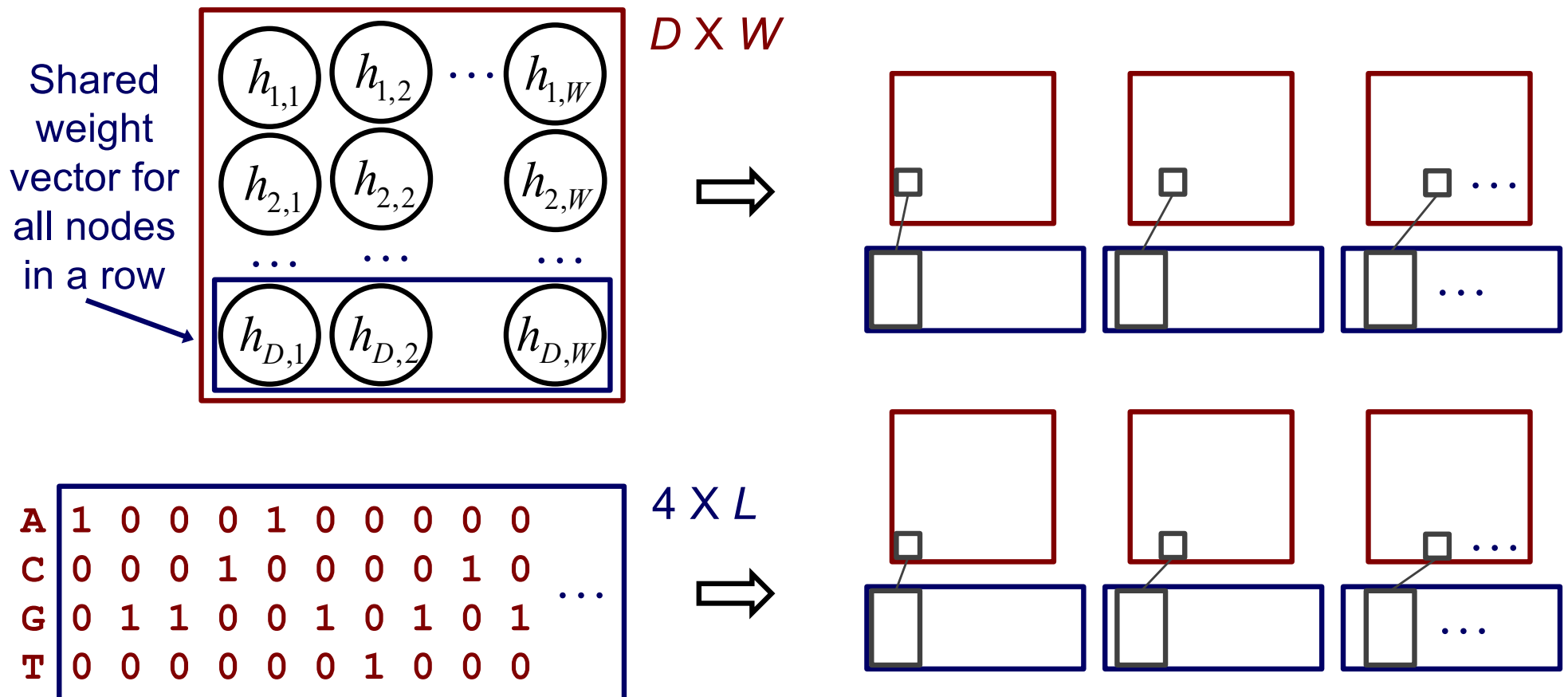
A G G C A G T G C G

First layer problems

- We already have a *lot* of parameters
 - Each hidden node has its own weight vector
- We're attempting to learn different motifs (filters) at each starting position

Convolutional layers

- Input sequence and hidden layer as matrices
- Share parameters for all hidden nodes in a row
 - Search for same motif at different starting positions

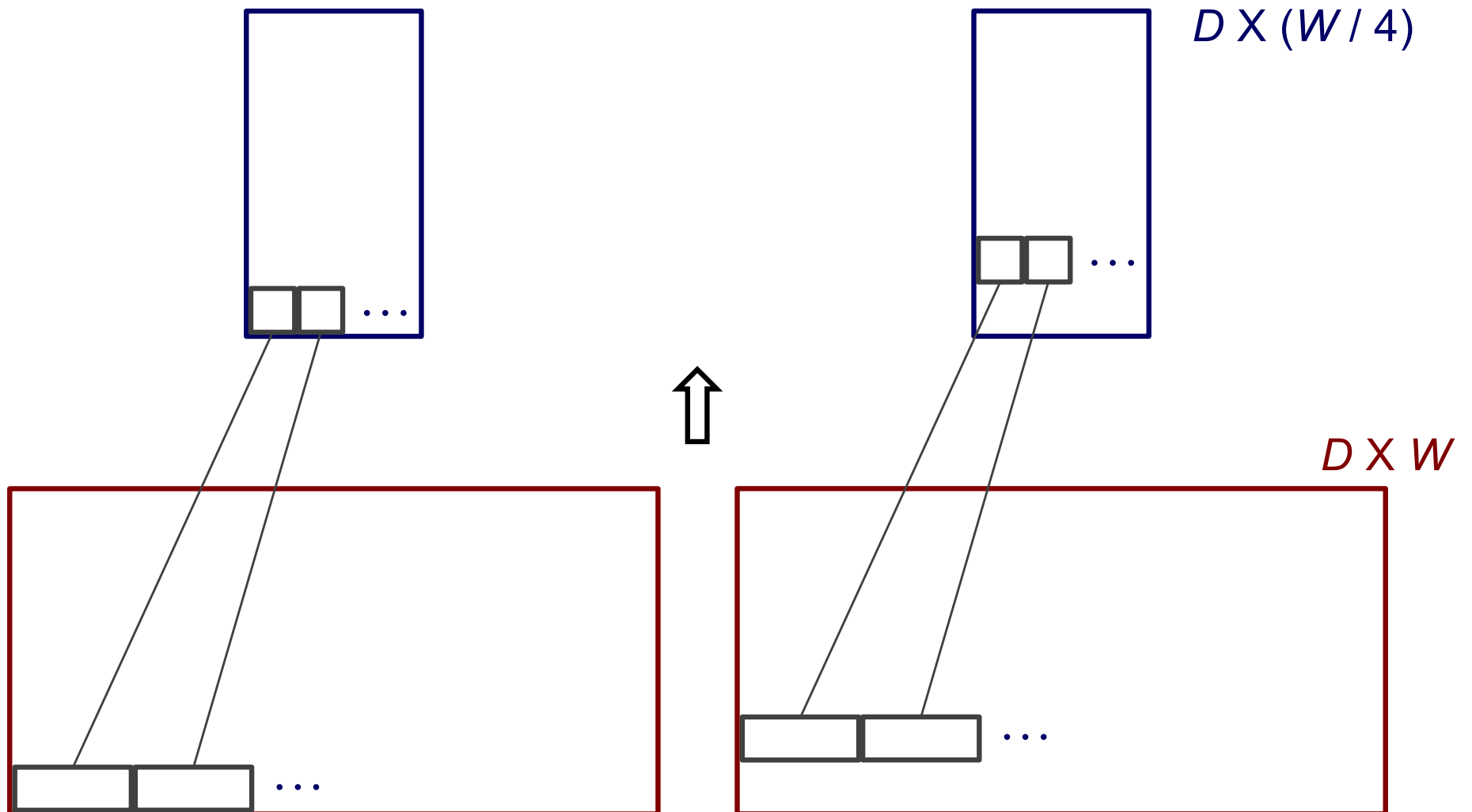


Pooling layers

- Account for sequence context
- Multiple motif matches in a *cis*-regulatory module
- Search for patterns at a higher spatial scale
 - Fire if motif detected anywhere within a window

Pooling layers

- Take max over window of 4 hidden nodes



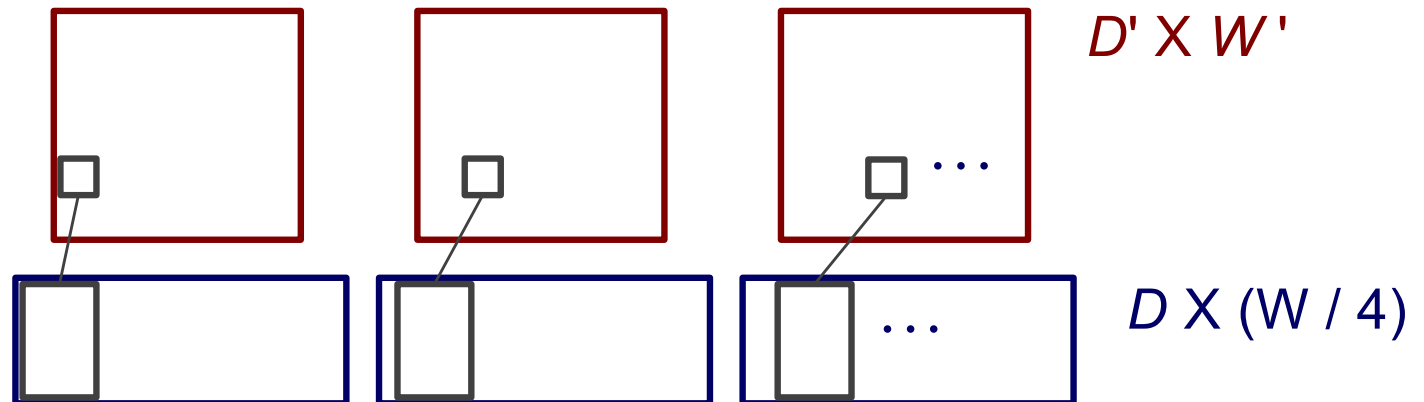
Subsequent hidden layers

- Next convolutional hidden layer on top of pooling layer

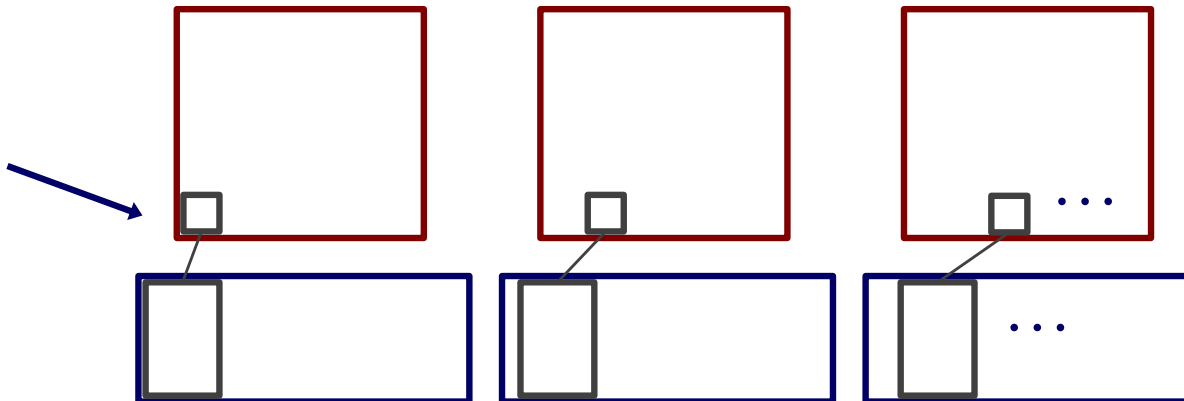
D' is new number of patterns

s' is new window size

$$W' = (W / 4) - s' + 1$$

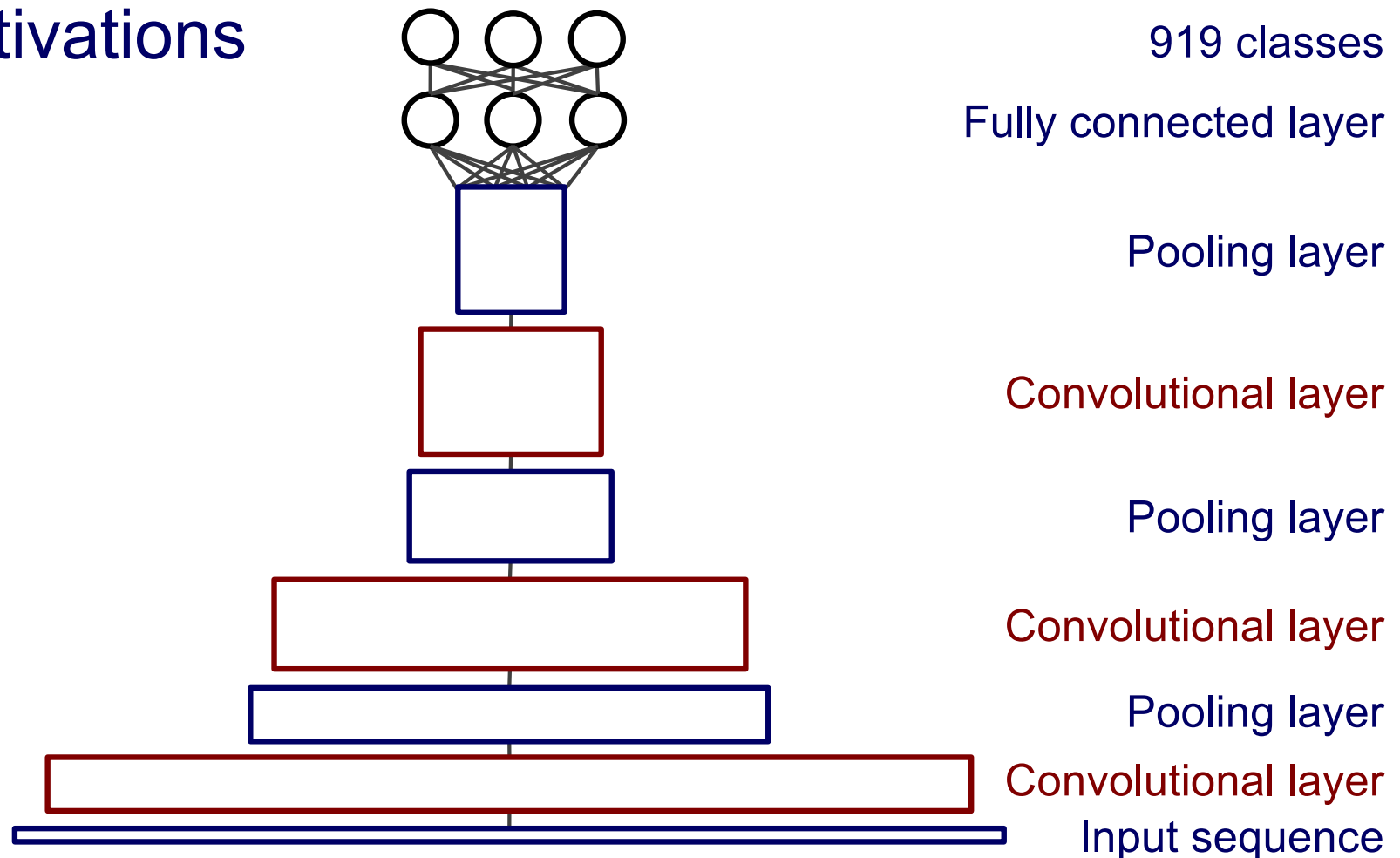


Once again,
shared weight
vector for all
nodes in a row



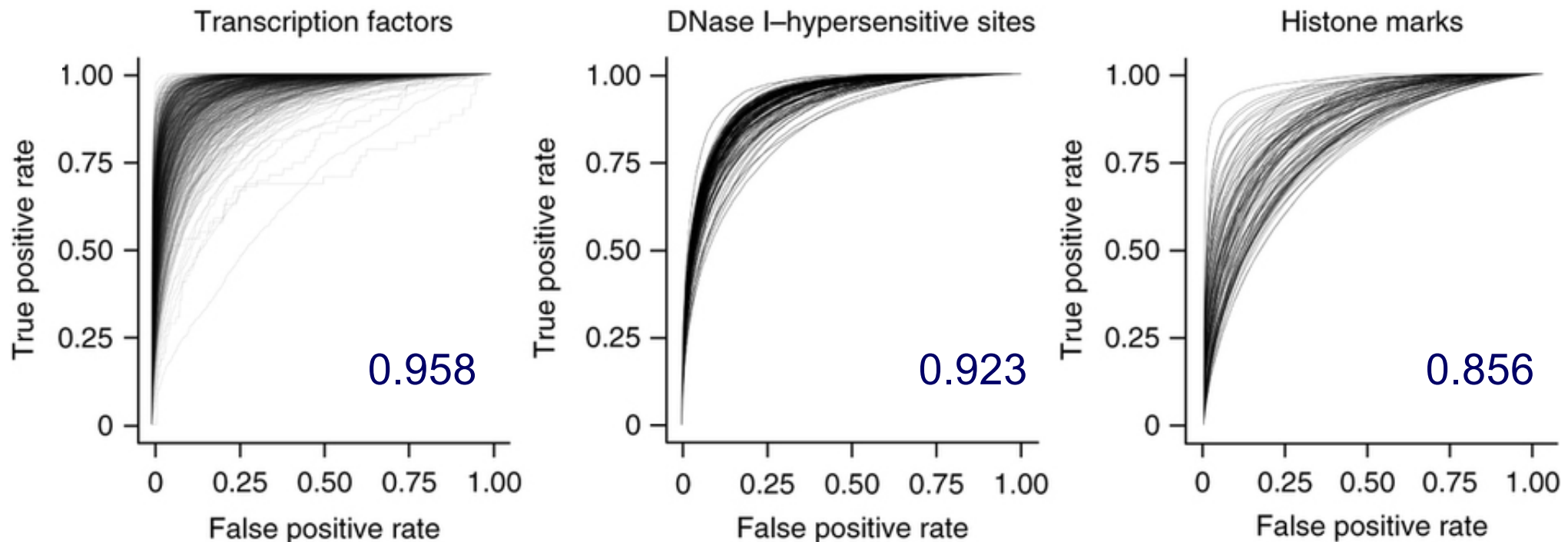
Full DeepSEA neural network

- Multitask output makes simultaneous prediction for each type of epigenetic data
- ReLU activations



Predicting epigenetic annotations

- Compute median AUROC for three types of classes



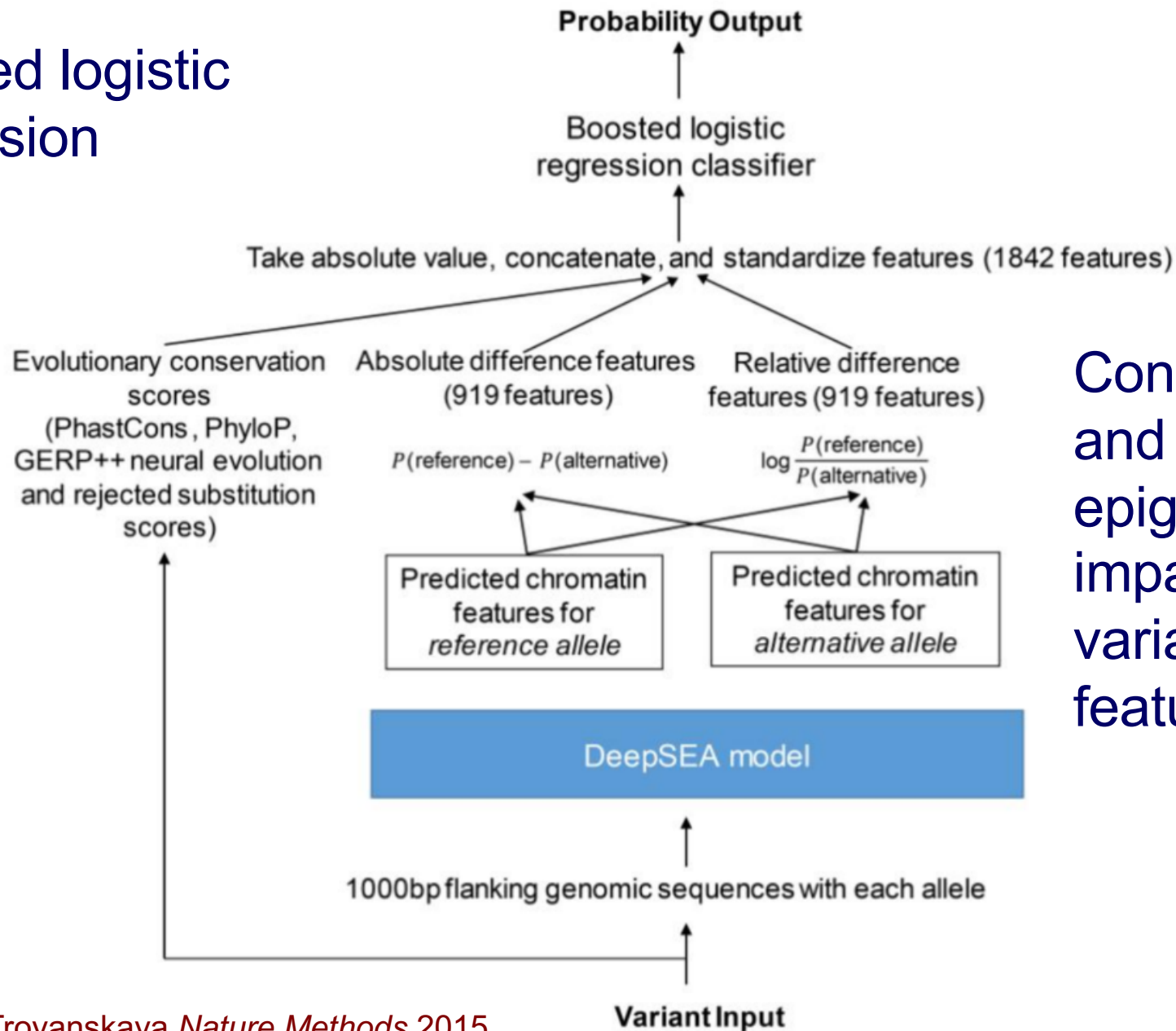
Zhou and Troyanskaya *Nature Methods* 2015

Predicting functional variants

- Can predict epigenetic signal for any novel variant (SNP, insertion, deletion)
- Define novel features to classify variant functionality
 - Difference in probability of signal for reference and alternative allele
- Train on SNPs annotated as regulatory variants in GWAS and eQTL databases

Predicting functional variants

Boosted logistic regression



Conservation and predicted epigenetic impact of variant as features

DeepSEA summary

- Ability to predict how unseen variants affect regulatory elements
- Accounts for sequence context of motif
- Parameter sharing with convolutional layers
- Multitask learning to improve hidden layer representations
- Does not extend to new types of cells and tissues
- AUROC is misleading for evaluating genome-wide epigenetic predictions

Predicting new TF-cell type pairs

Training data

TF	Cell type
A	1
A	2
A	3
B	3
B	4
C	1
C	4

- DeepSEA cannot predict pairs not present in training data
 - Can predict TF A in cell type 1
 - Not TF A in cell type 4
- New methods can
 - TFImpute
 - FactorNet
 - Virtual ChIP-seq

Deep learning is rampant in biology and medicine

- Network interpretation: DeepLIFT
- Protein structure prediction
- Cell lineage prediction
- Variant calling
- Model zoo: Kipoi
- Comprehensive review: deep-review