

Deep Learning Applications

BMI/CS 776

www.biostat.wisc.edu/bmi776/

Spring 2022

Daifeng Wang

daifeng.wang@wisc.edu

Goals for lecture

Key concepts

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

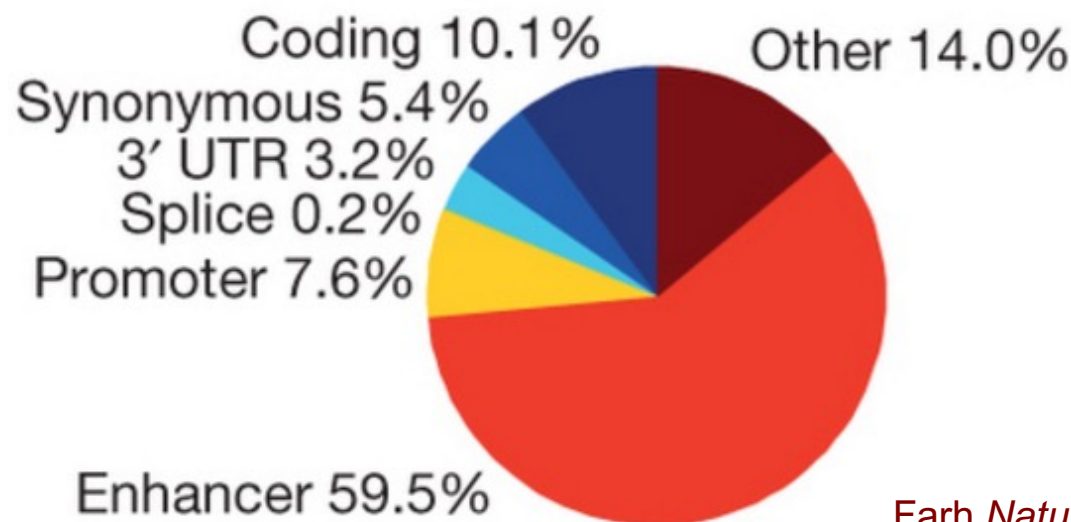
GWAS output

- 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

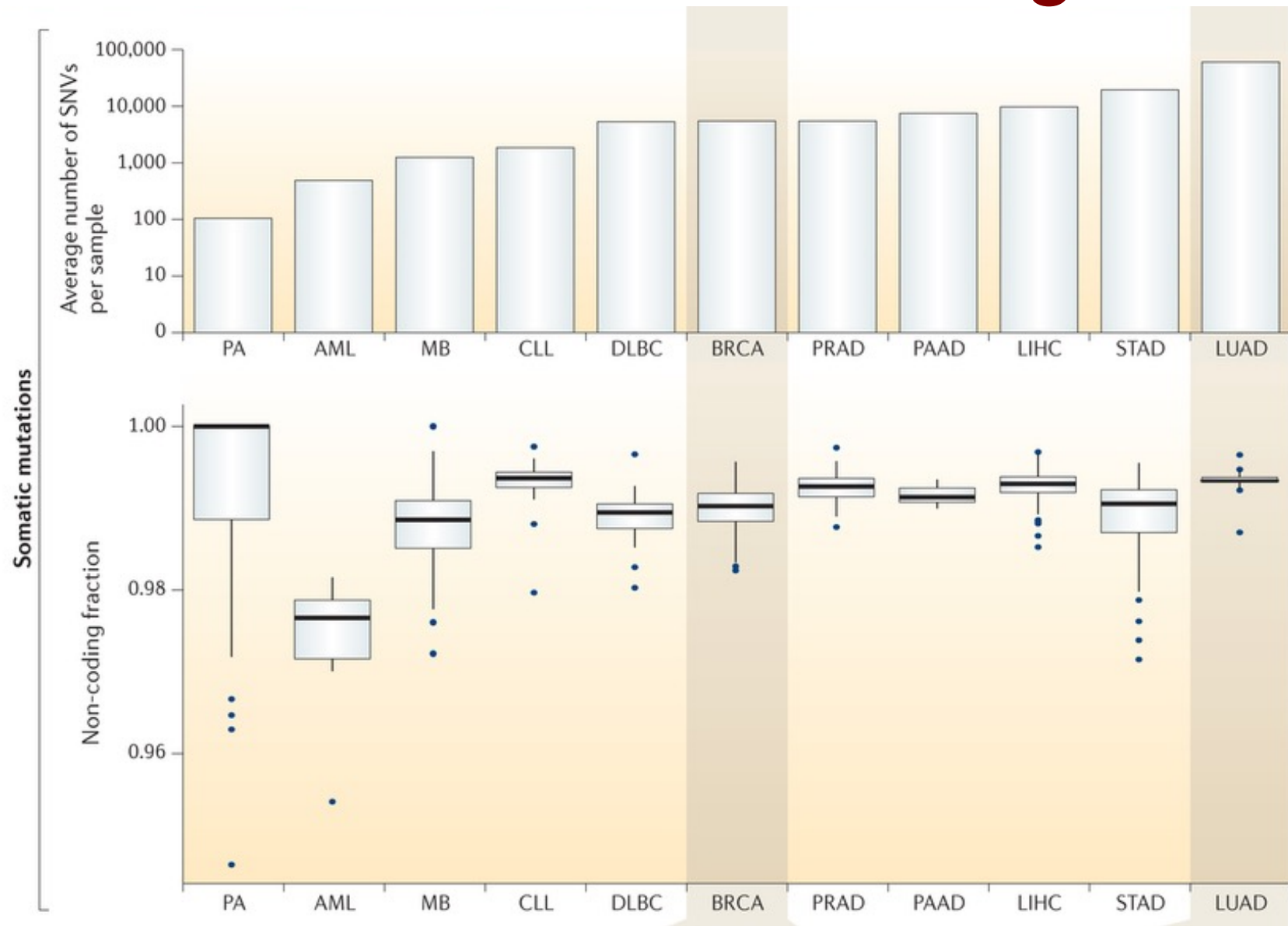
- Meta-analysis of GWAS for 21 autoimmune diseases
 - Rheumatoid arthritis, lupus, multiple sclerosis, etc.
- Method to prioritize candidate causal SNPS
- **90%** of causal variants are noncoding

GWAS SNPs



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

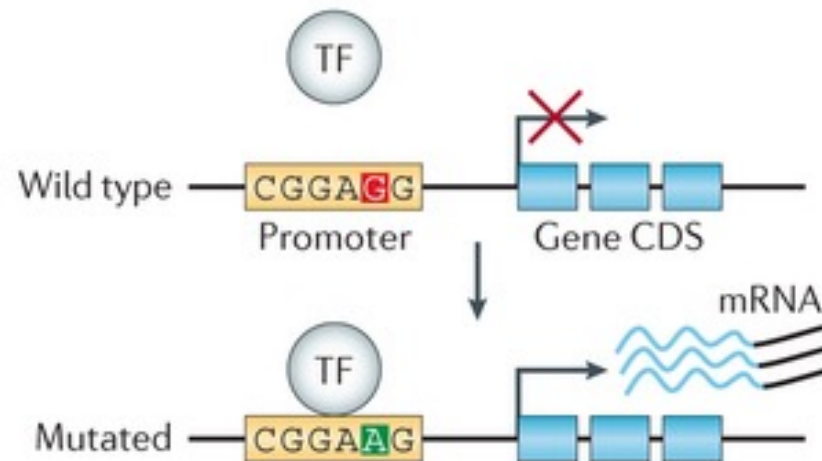
Ways a noncoding variant can be functional

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

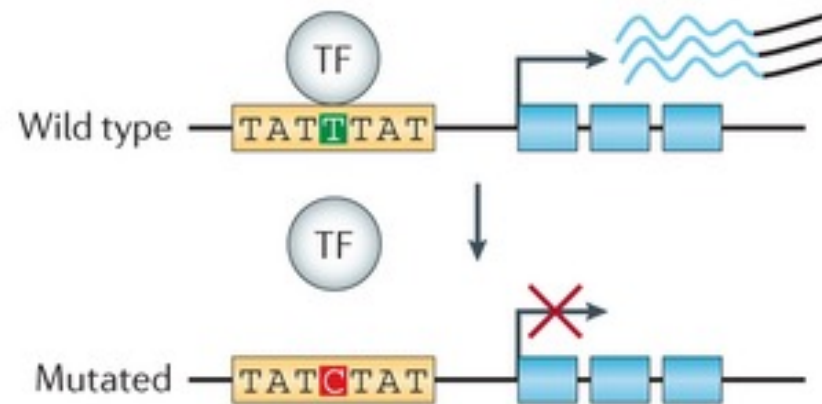
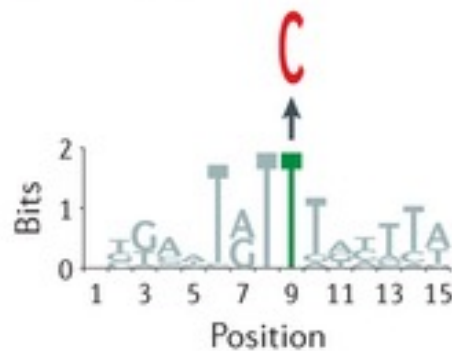
Examples in Ward and Kellis *Nature Biotechnology* 2012

Variants altering motifs

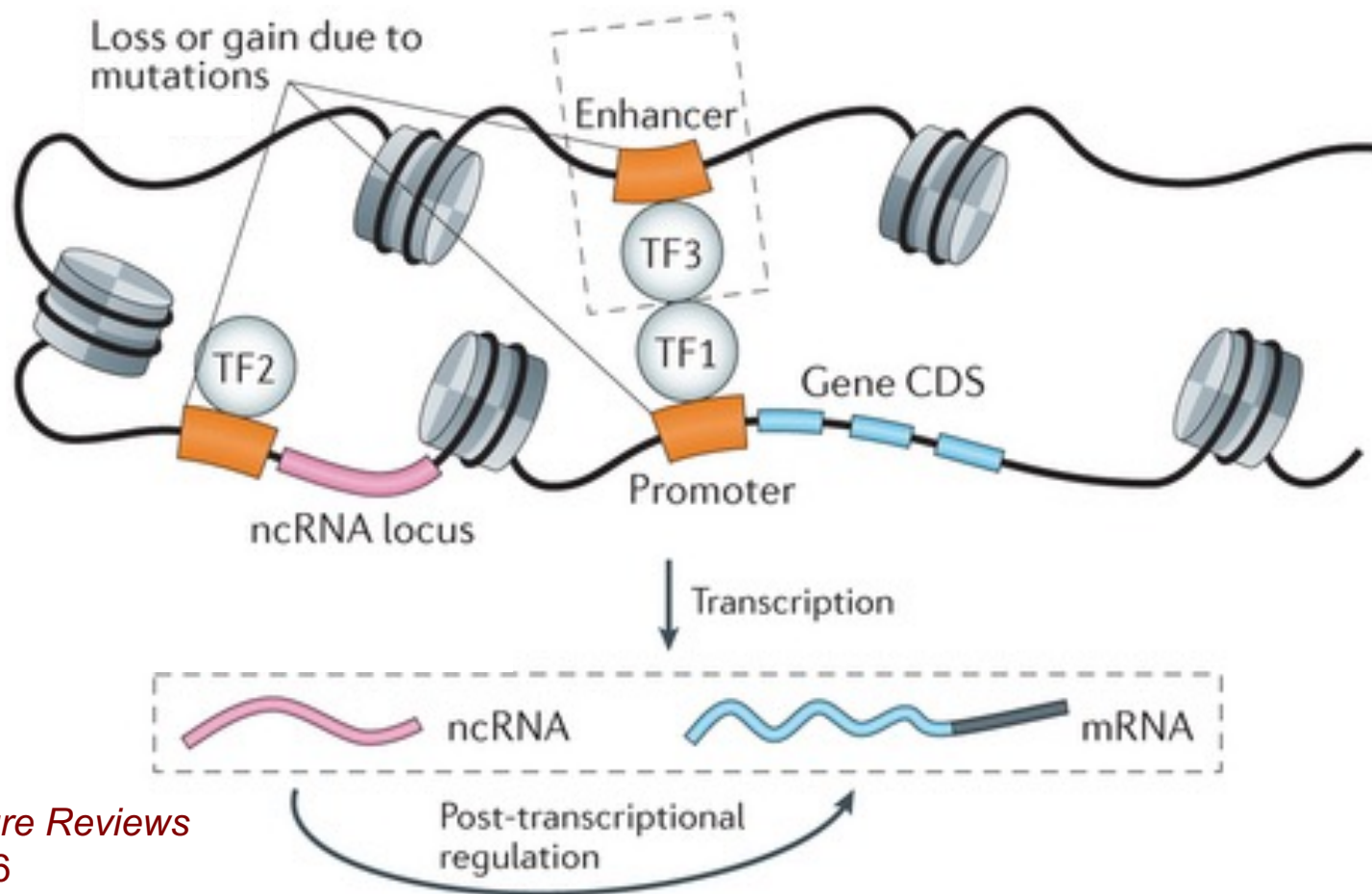
Gain of motif



Loss of motif



Variants affect proximal and distal regulators

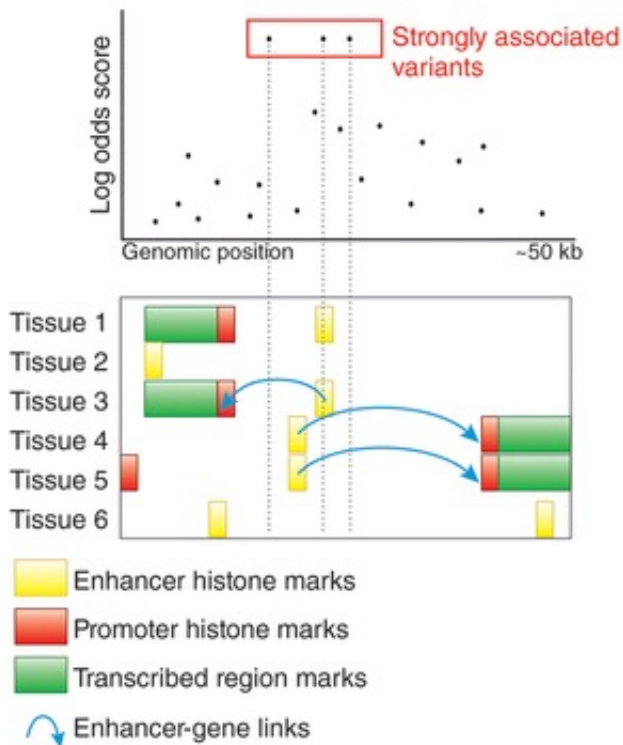


Khurana *Nature Reviews Genetics* 2016

Evidence used to prioritize noncoding variants

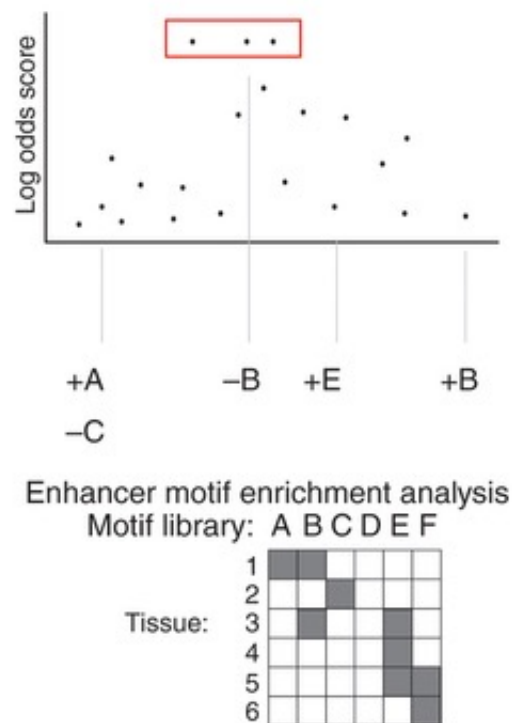
Interpreting GWAS signals using functional and comparative genomics datasets

a Dissect associated haplotype using functional genomics



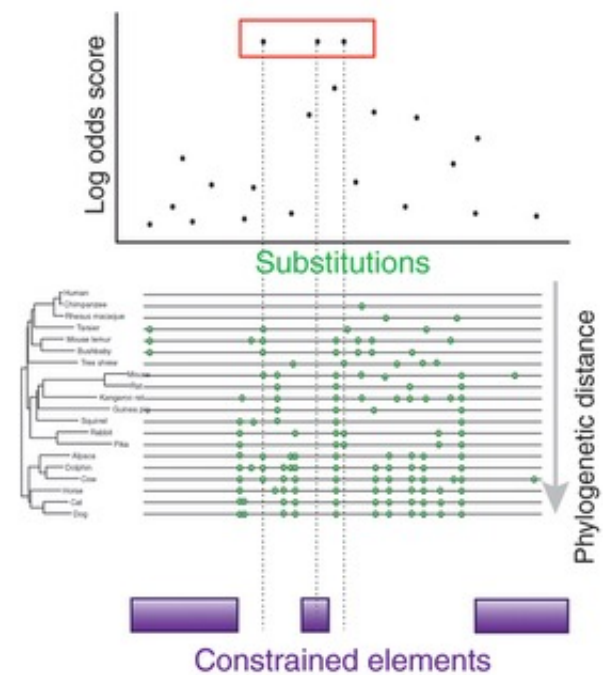
Chromatin state annotations

b Dissect associated haplotype using regulatory genomics



Motifs altered by variants

c Dissect associated haplotype using comparative genomics



Mammalian constraint

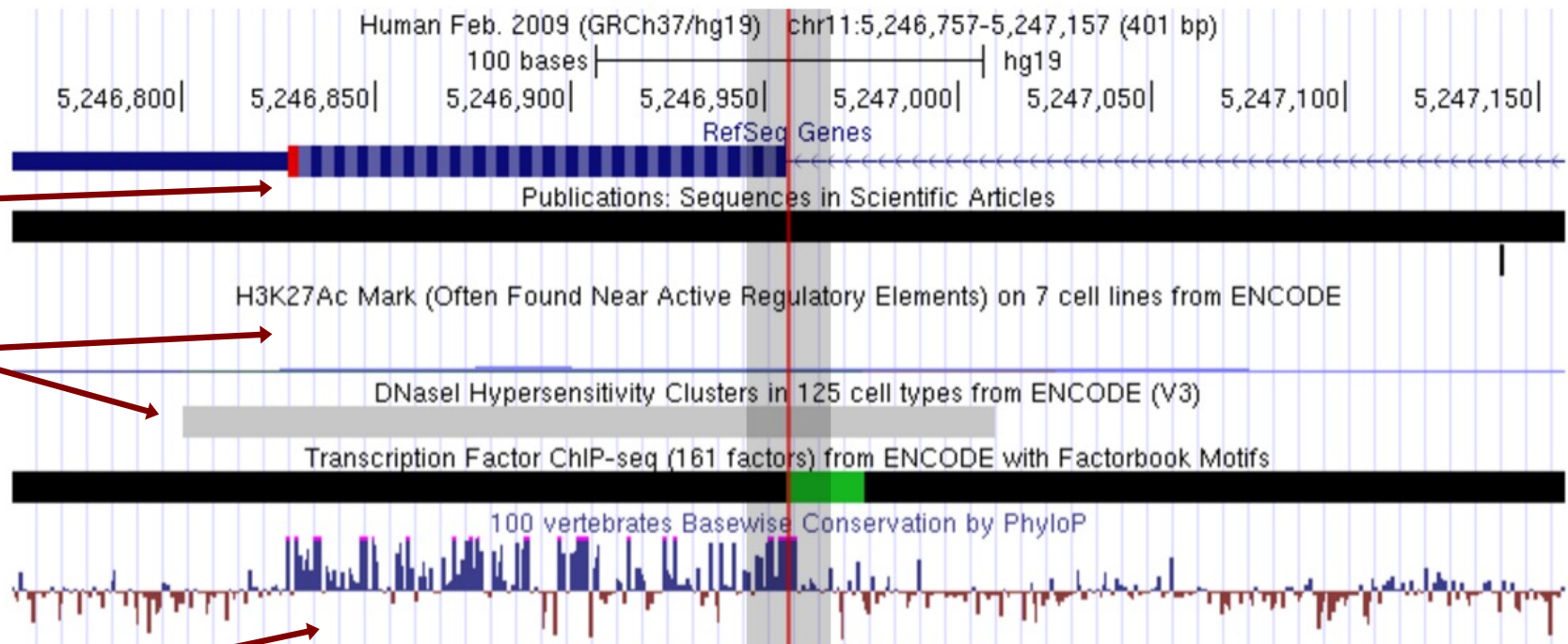
Visualizing evidence



Data supporting chr11:5246957 (rs33914668)

Summary of evidence → **Score: 2a**

Likely to affect binding



Genes →

Epigenetic annotations →

Conservation →

Affected motifs →



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

Kircher *Nature Genetics* 2014

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

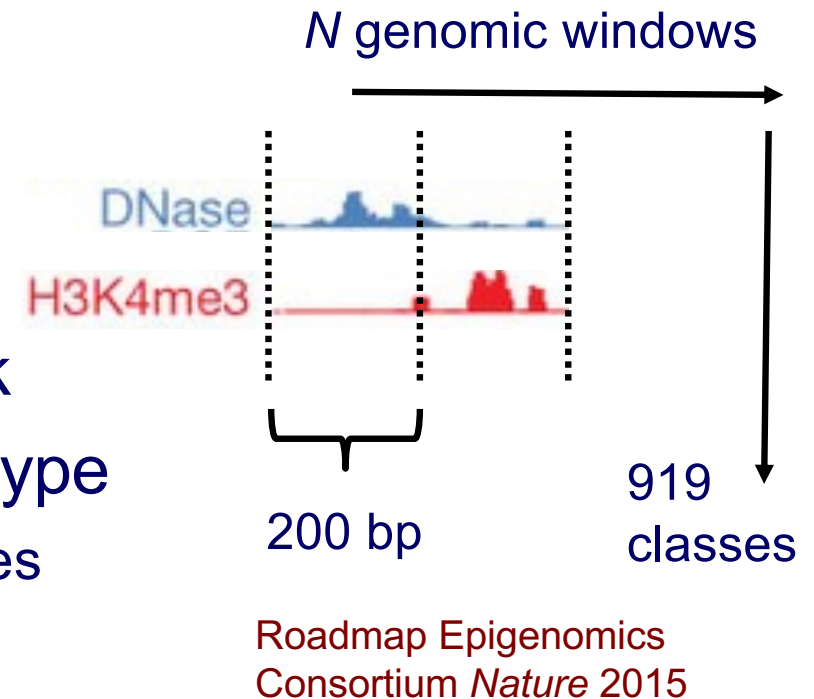
DeepSEA

- 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

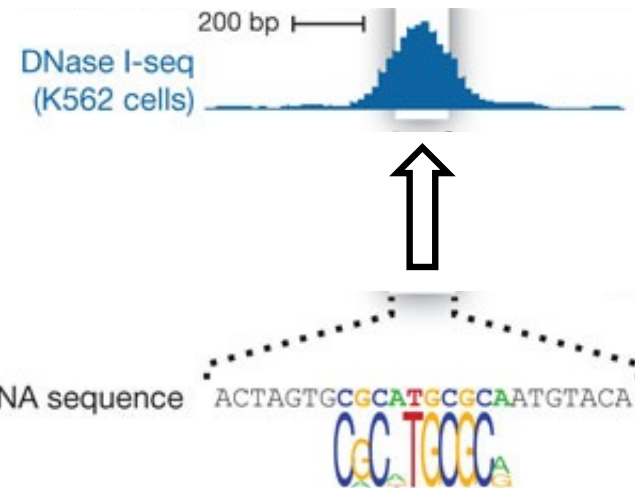
$$x_i =$$

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

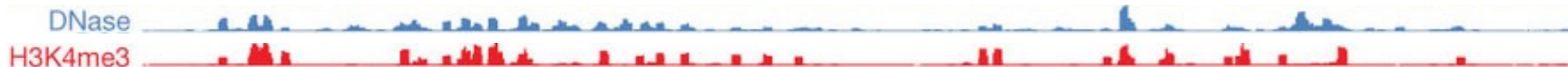
- Learn preferences of DNA-binding proteins
 - Locally: “motifs” and other simple sequence patterns
 - Sequence context: “*cis*-regulatory modules”

- Support nonlinear decision boundaries



Neph *Nature* 2012

- Multiple, related prediction tasks

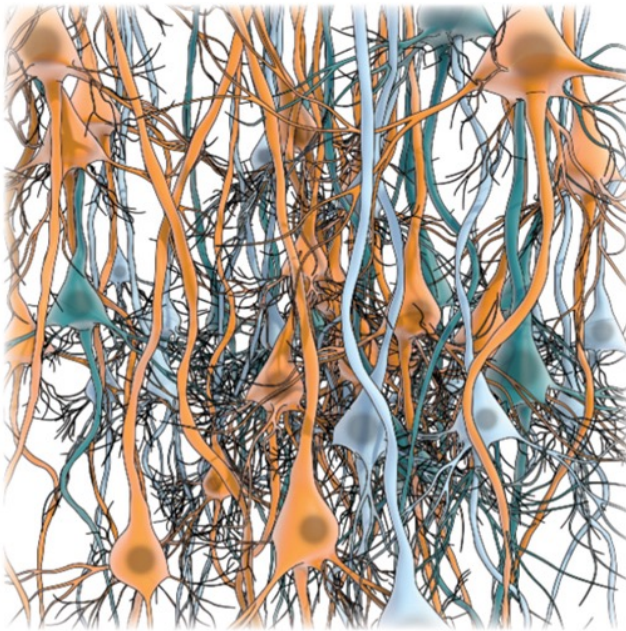


Roadmap Epigenomics Consortium *Nature* 2015

Neuroscience to artificial intelligence

Brain circuitry and learning

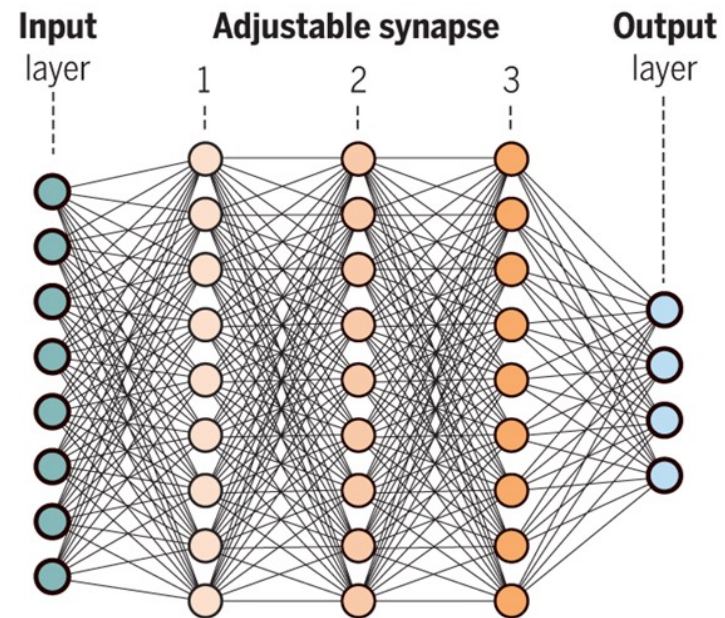
A major open question is whether the highly simplified structures of current network models compared with cortical circuits are sufficient to capture the full range of human-like learning and cognition.



Complex neural network

Connectivity in cortical networks includes rich sets of connections, including local and long-range lateral connectivity, and top-down connections from high to low levels of the hierarchy.

Shimon Ullman, *Science* 15 Feb 2019:
Vol. 363, Issue 6428, pp. 692-693

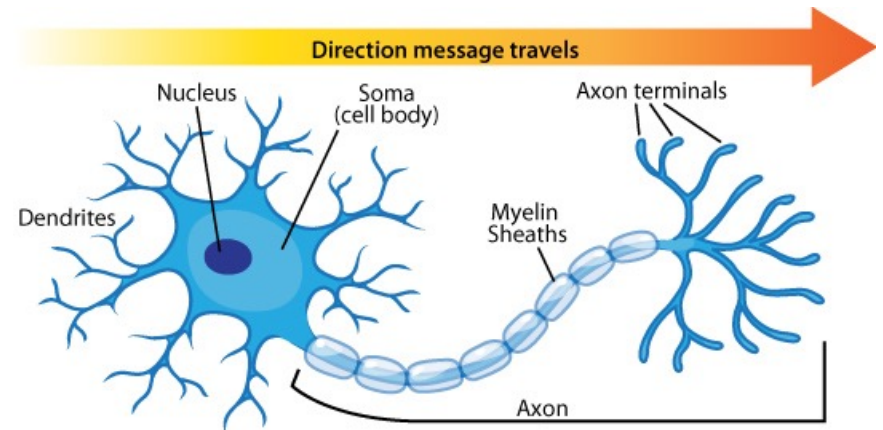


Informed AI network

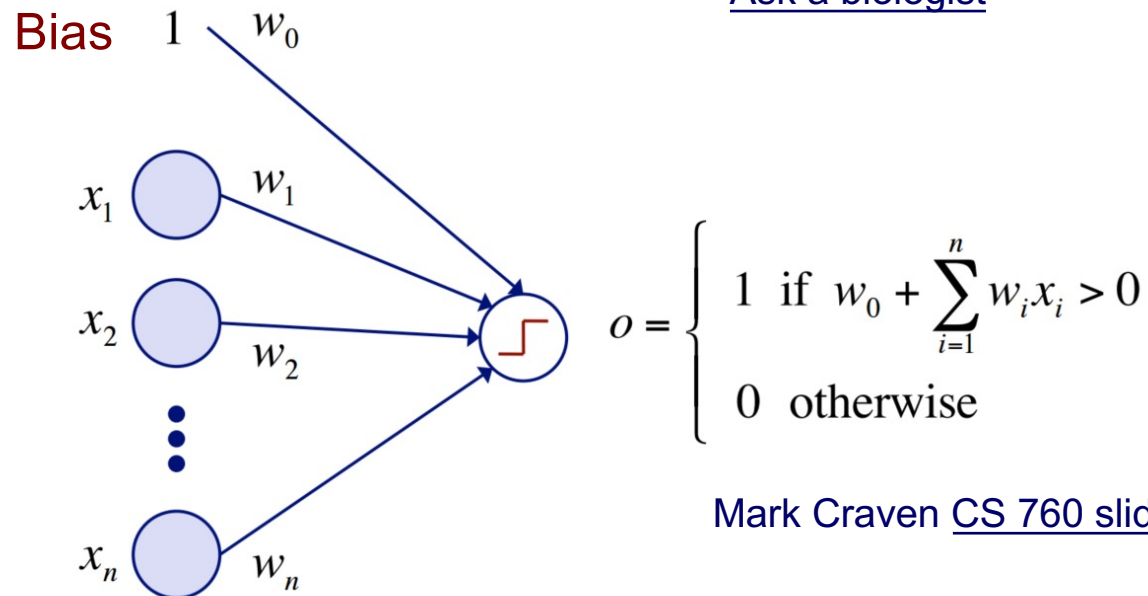
Biological innate connectivity patterns provide mechanisms that guide human cognitive learning. Discovering similar mechanisms, by machine learning or by mimicking the human brain, may prove crucial for future artificial systems with human-like cognitive abilities.

Perceptron

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



Ask a biologist



Mark Craven CS 760 slides

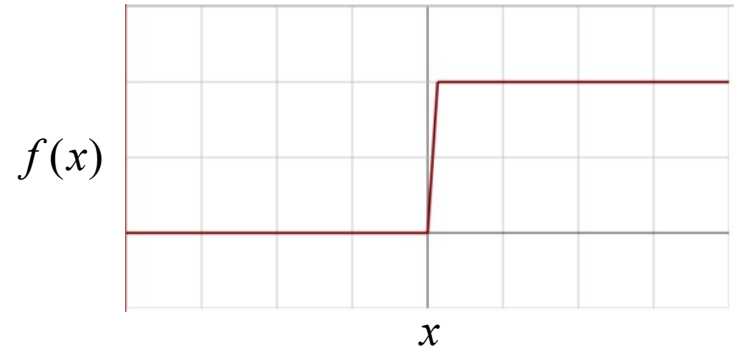
$$x_{1,A} \dots x_{1000,T}$$

Activation function

- What makes the neuron “fire”?

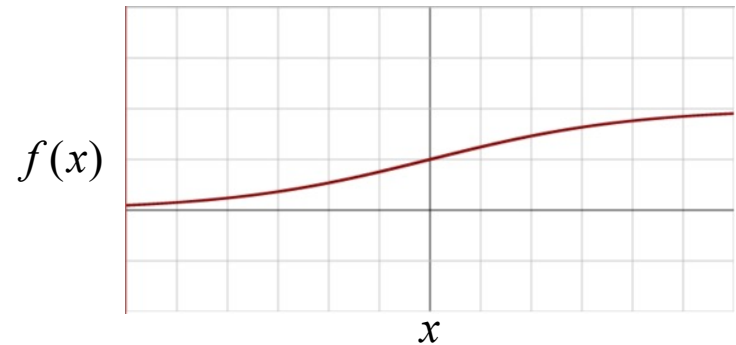
- Step function

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



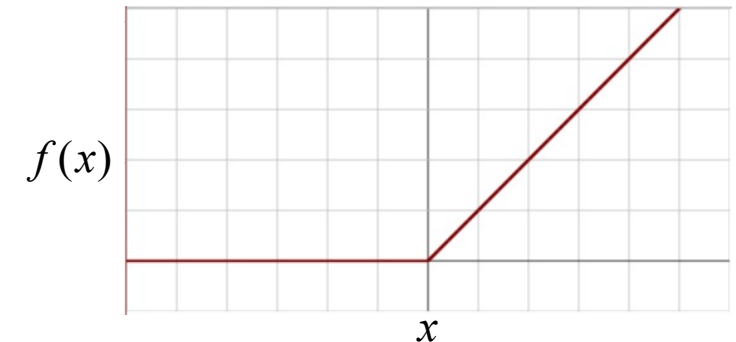
- Sigmoid function

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



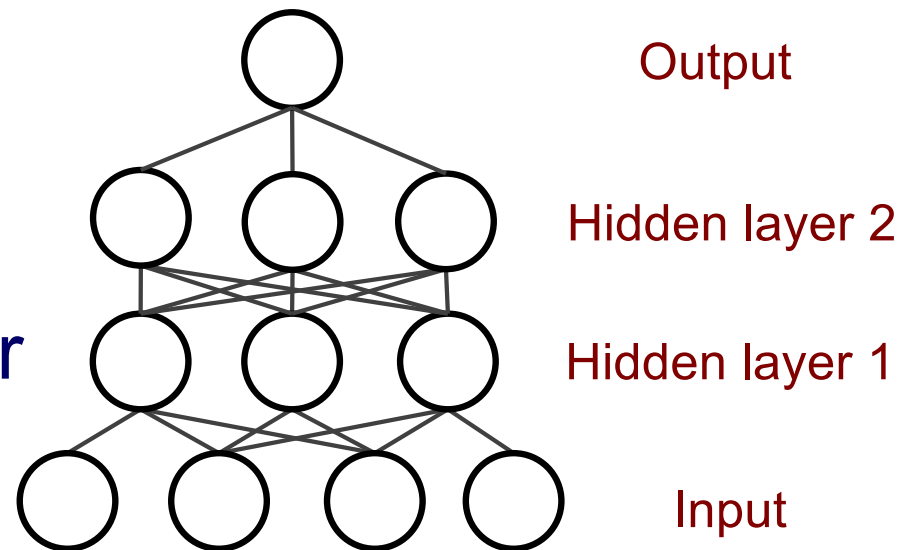
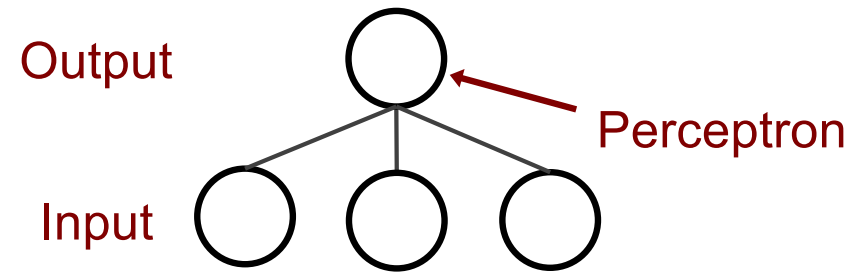
- Rectified linear unit (ReLU)

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

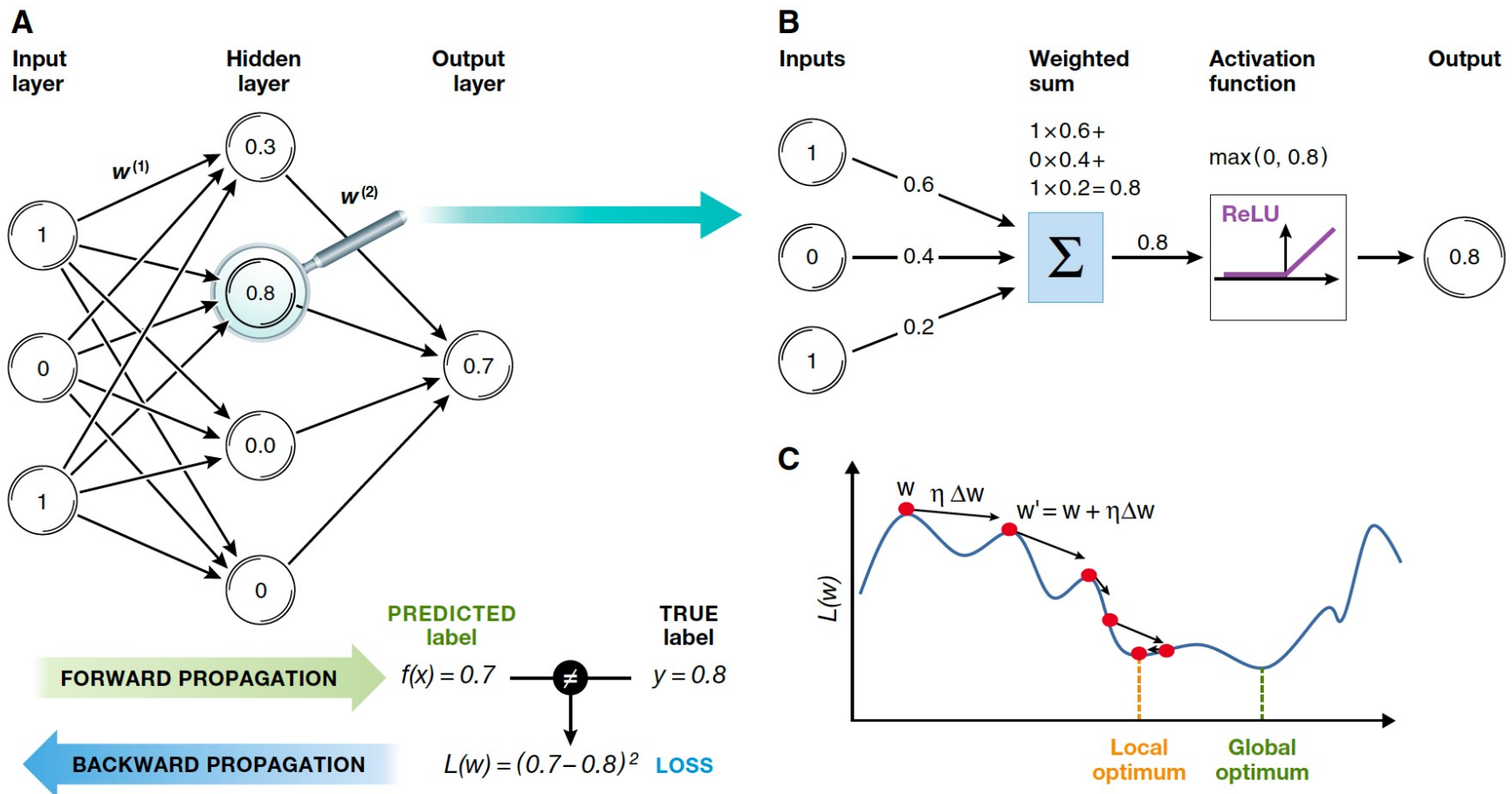


Neural networks

- 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](#)



Train NN with backpropagation

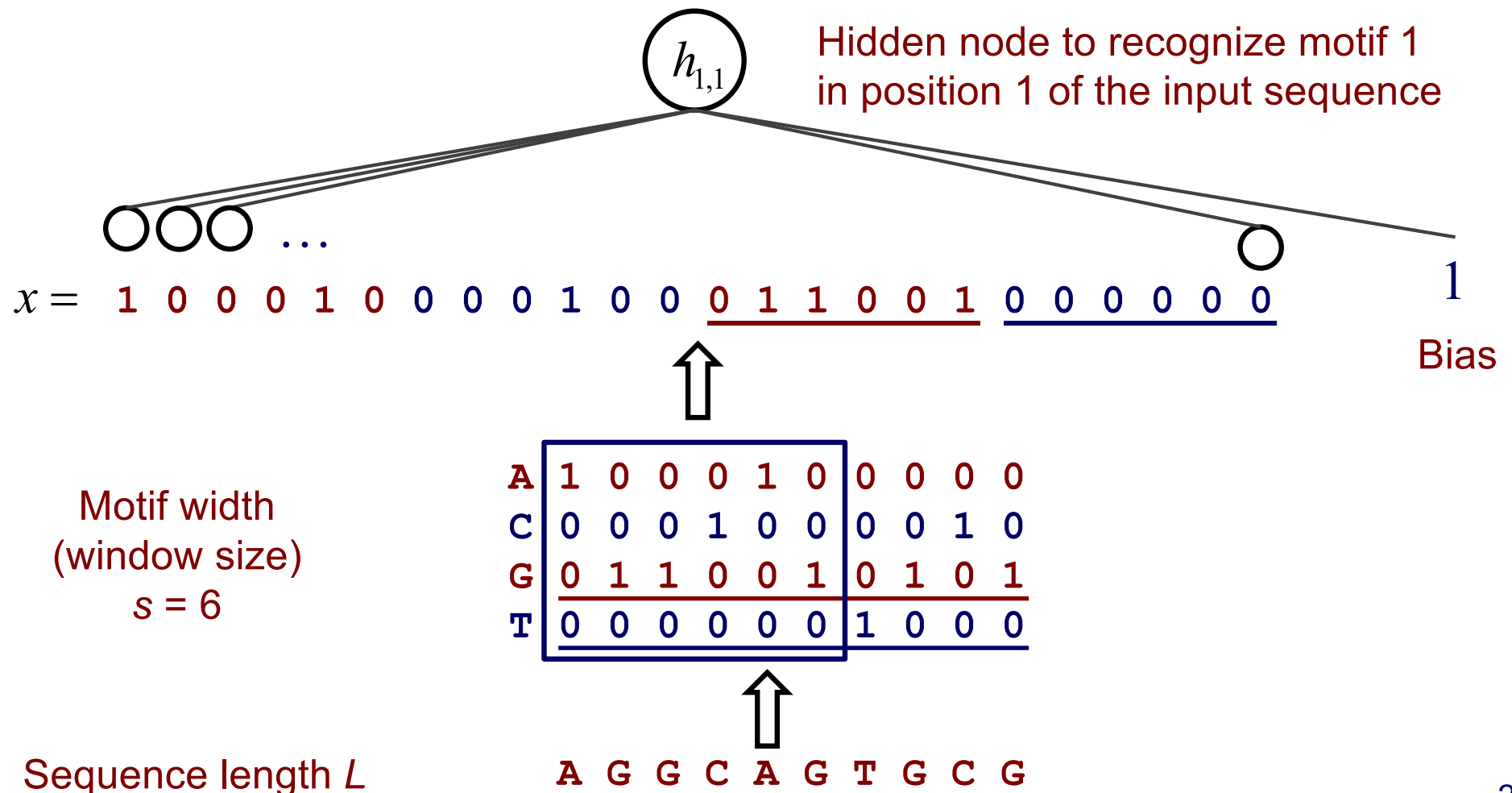


Neural network examples

- 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

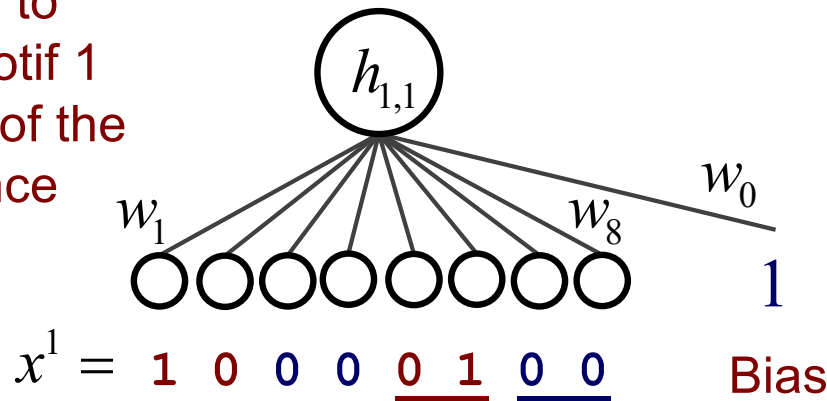
- First hidden layer scans input sequence
- Activation function fires if “motif” is recognized



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	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

**Input
sequence 1**

A G G C A G T G C G

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

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

e.g., from
PWM of a
motif

Weights times input

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

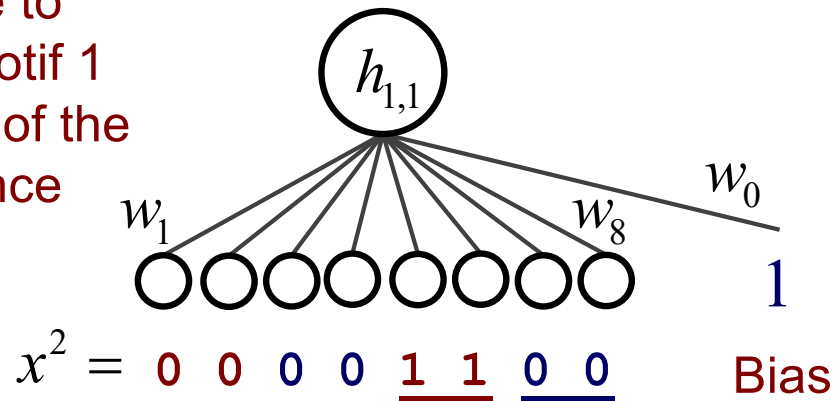
Apply ReLU activation function

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

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



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

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

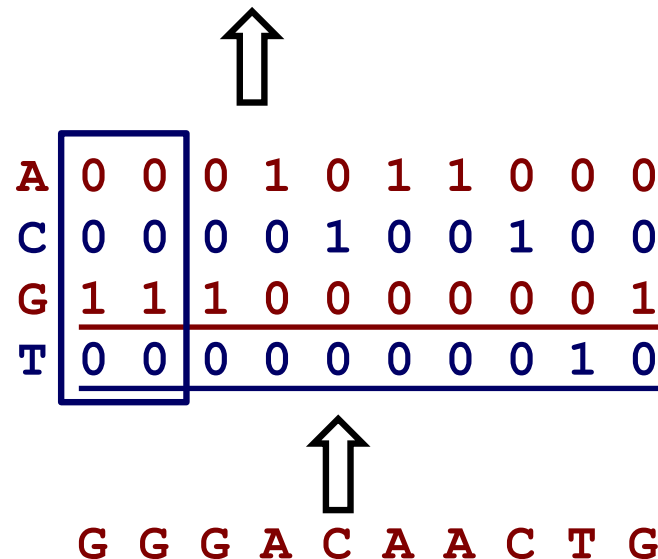
Weights times input

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

Apply ReLU activation function

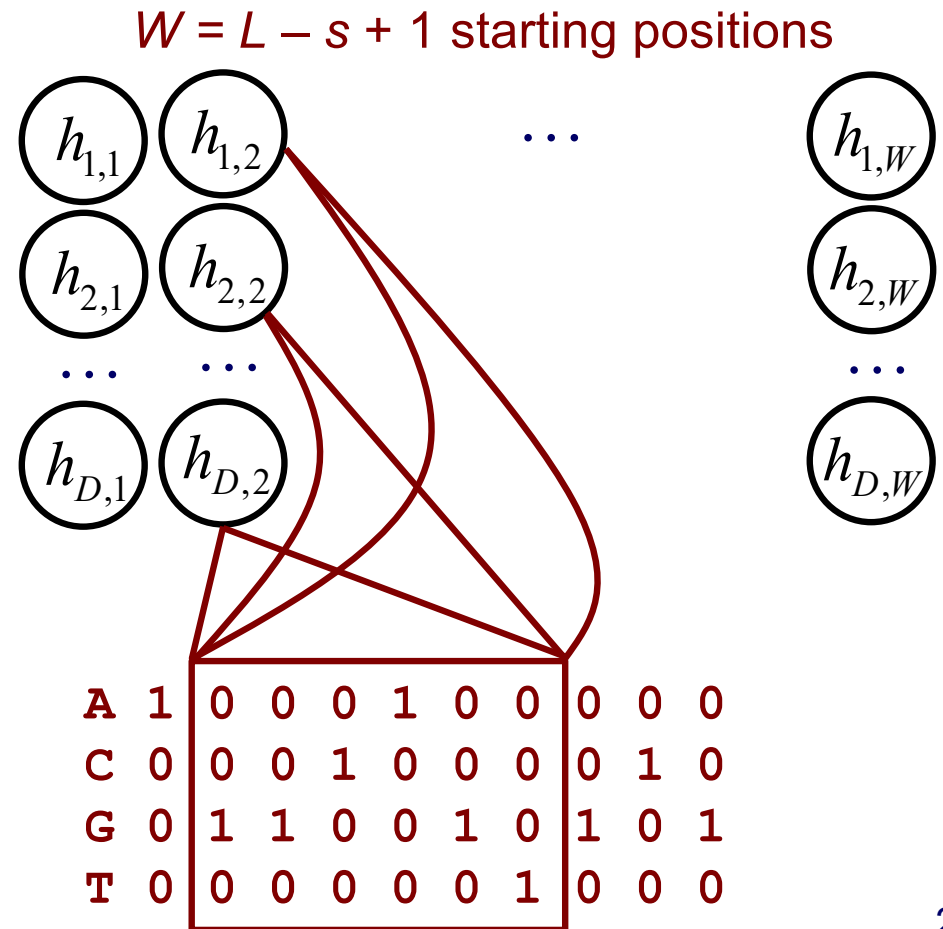
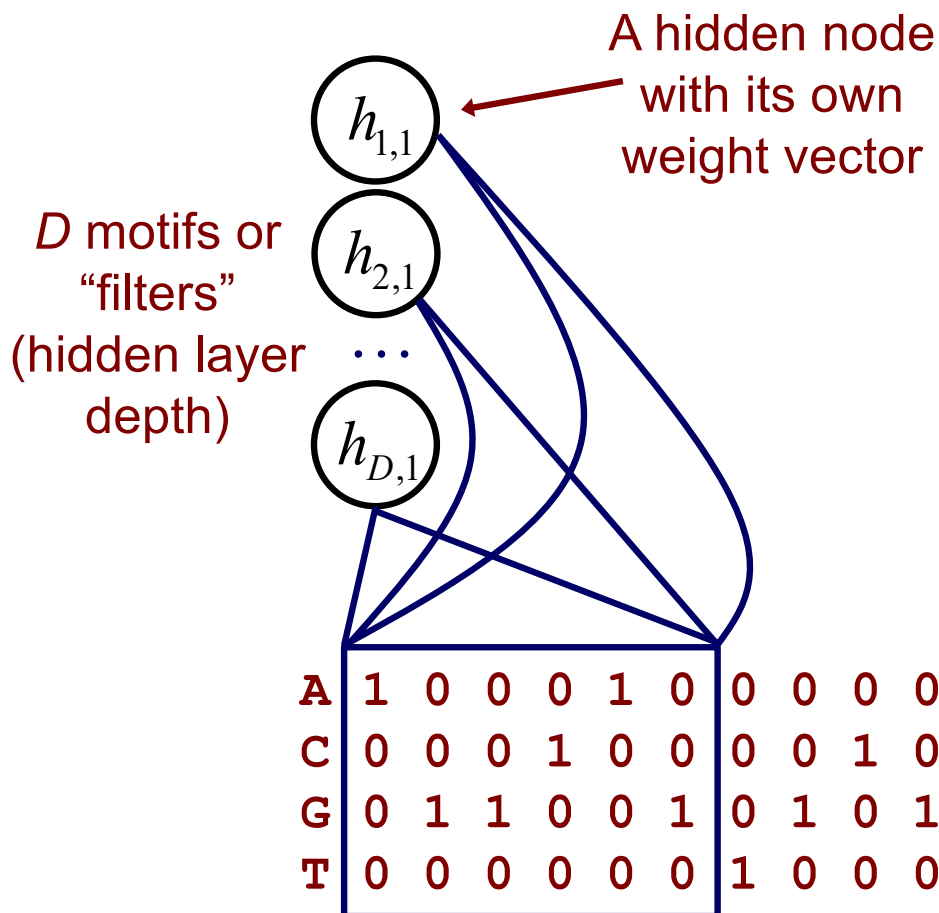
$$\max(0, -0.2) = 0$$

Input sequence 2



First hidden layer

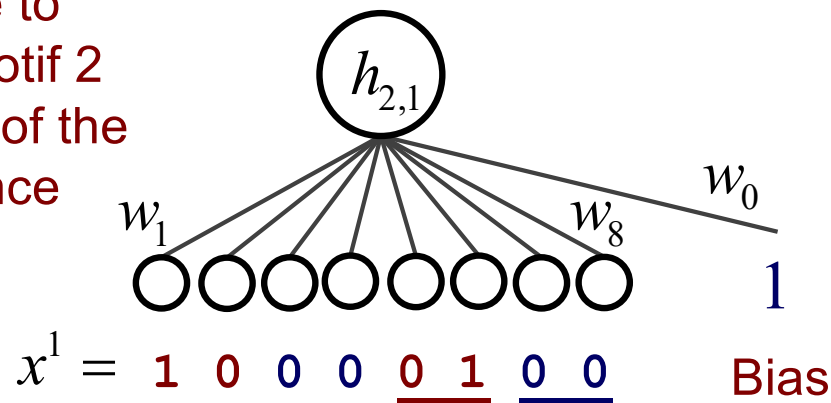
- Multiple hidden nodes to recognize different motifs at a particular position
- Check for motif at each position in sequence



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

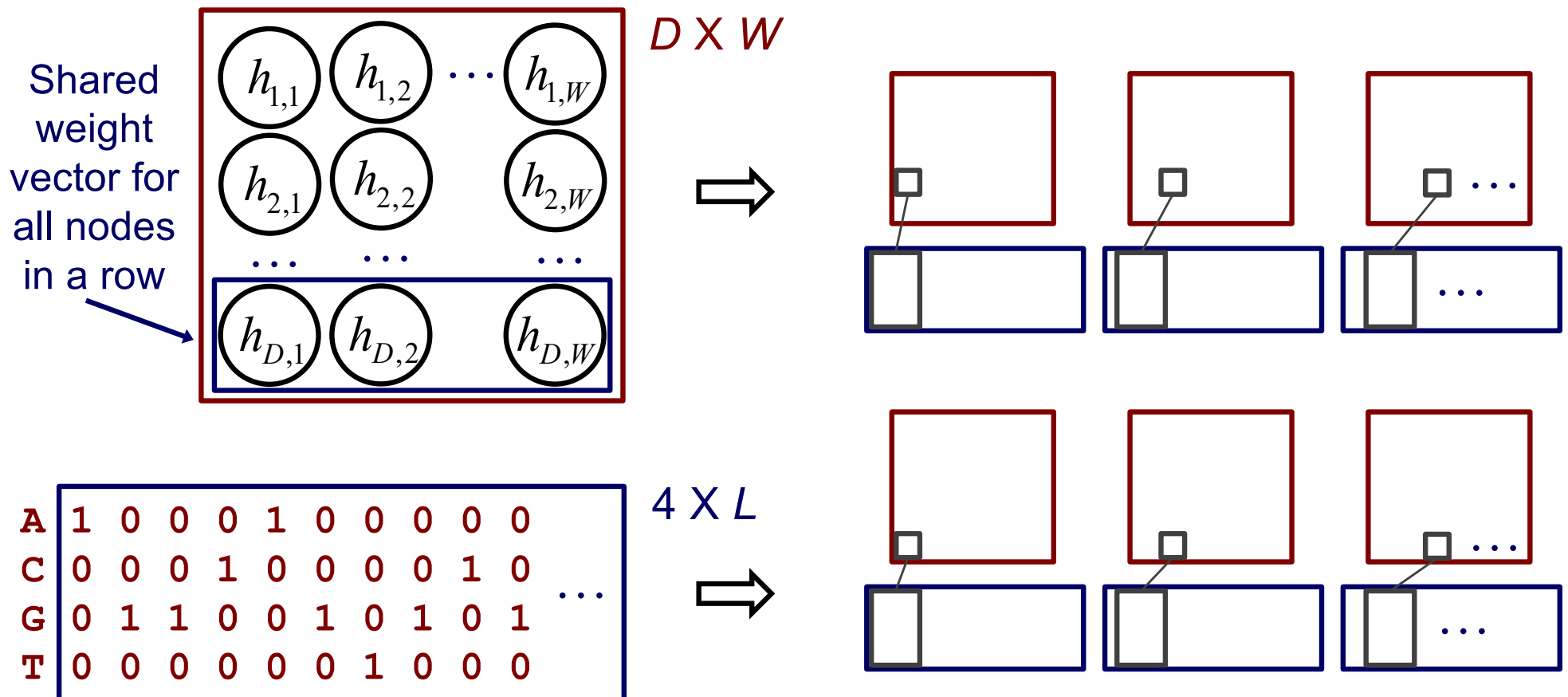
A G G C A G T G C G

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
 - $D*W$ hidden nodes

Convolutional layers

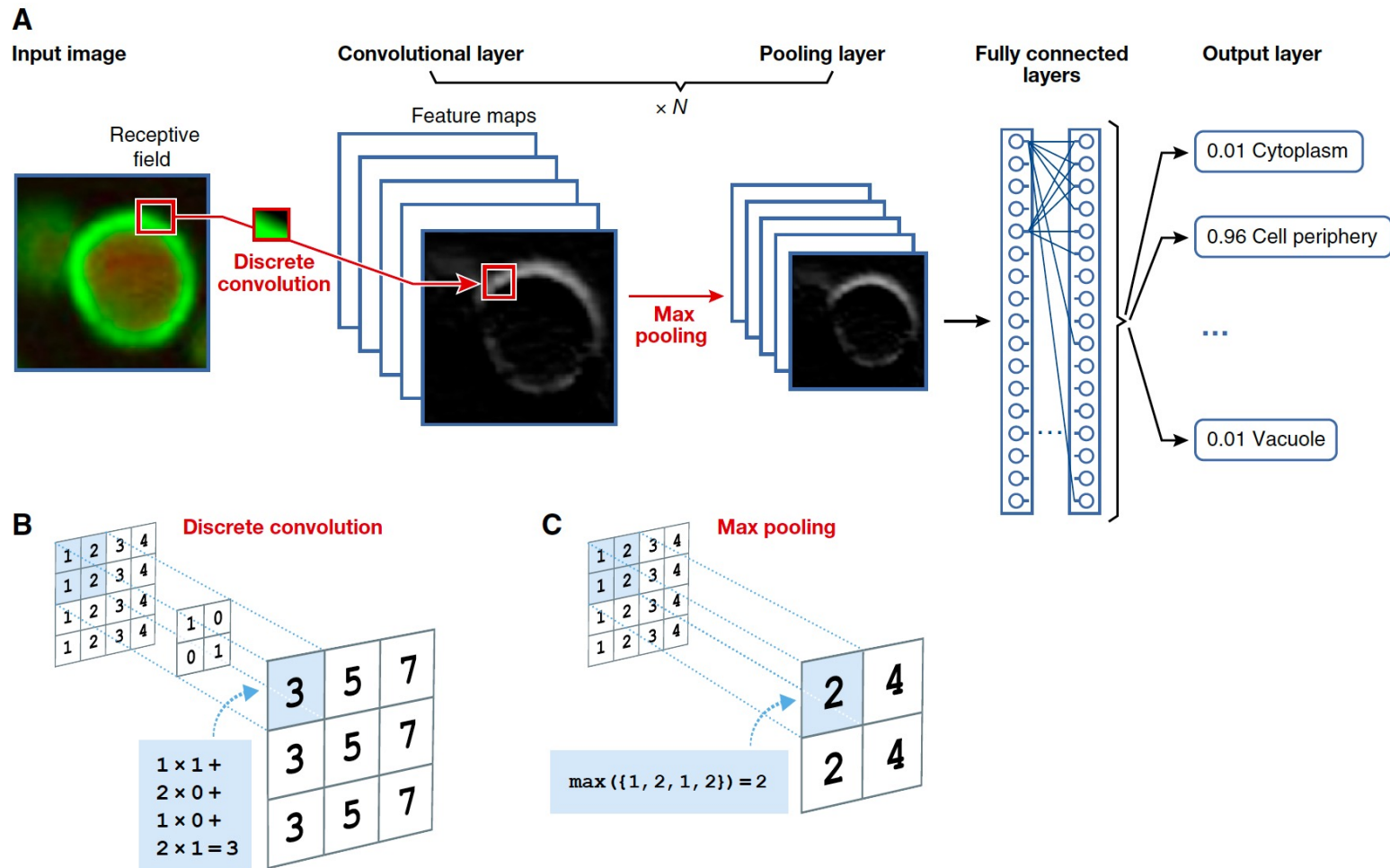
- Input sequence and hidden layer as matrices
- Share parameters for all hidden nodes in a row
 - Search for same motif at different starting positions



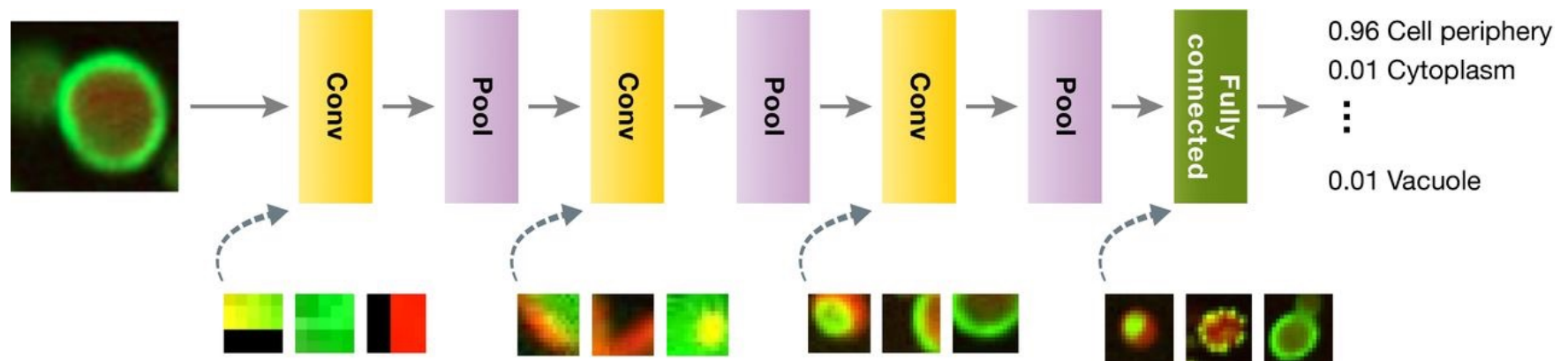
Pooling layers

- 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

Convolution and Pooling



Convolution and pooling operators are stacked, thereby creating a deep network for image analysis



Christof Angermueller et al. Mol Syst Biol 2016;12:878

A pre-trained network can be used as a generic feature extractor

First layer features

	 In top left?	 In top right?	...	 In bottom right?
	0.21	0.24		0.01
	0.02	0.01		0.25
	0.01	0.03		0.19

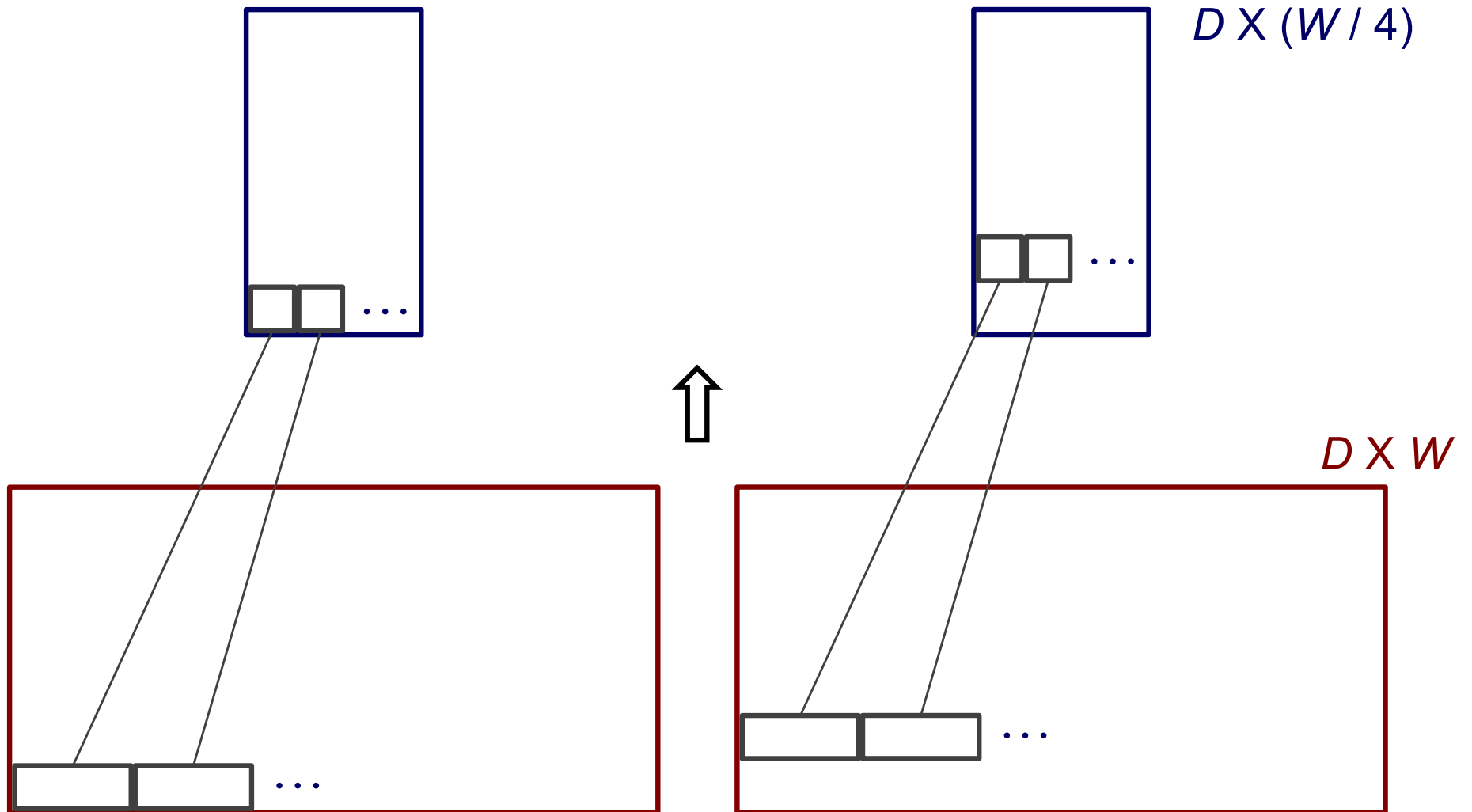
Third layer features

	 In left?	 In right?	...	 In bottom?
	2.51	0.02		2.92
	0.03	0.01		0.02
	0.02	0.01		0.01

Christof Angermueller et al. Mol Syst Biol 2016;12:878

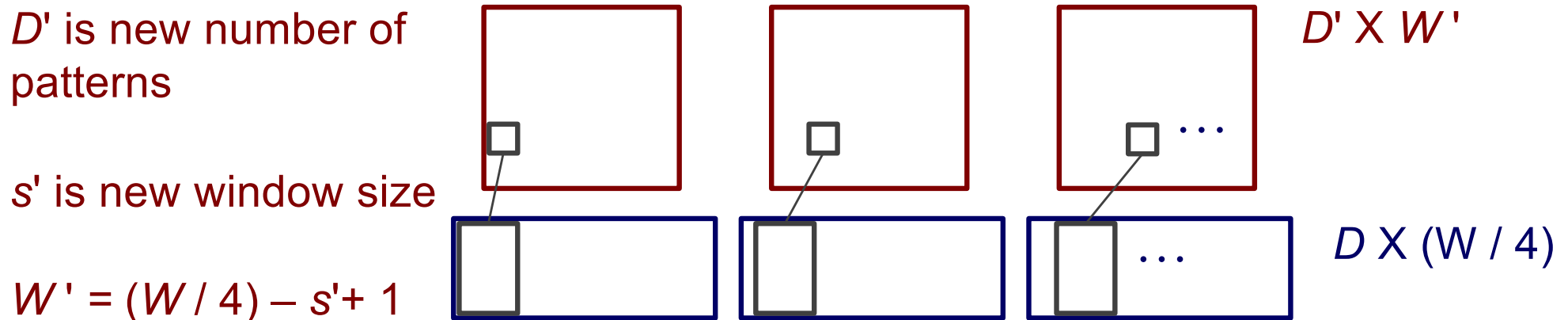
Pooling layers

- Take max over window of 4 hidden nodes

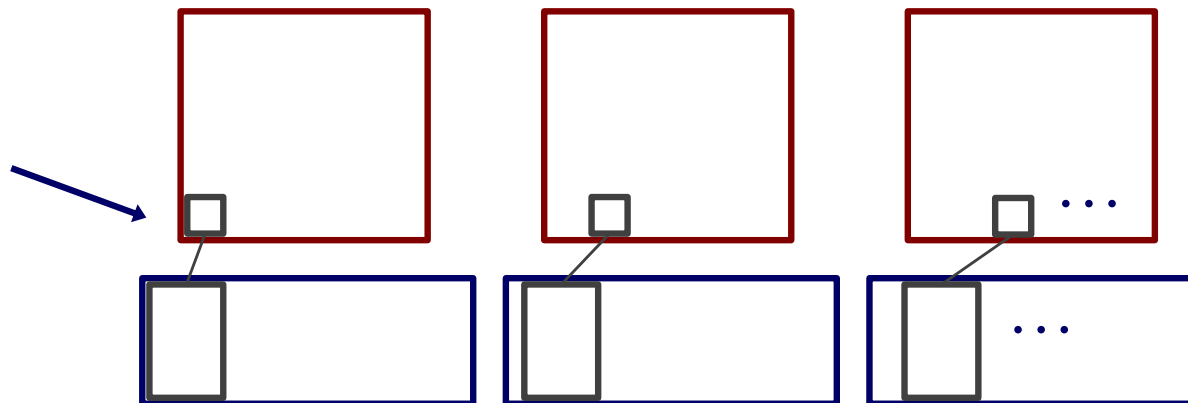


Subsequent hidden layers

- Next convolutional hidden layer on top of pooling layer

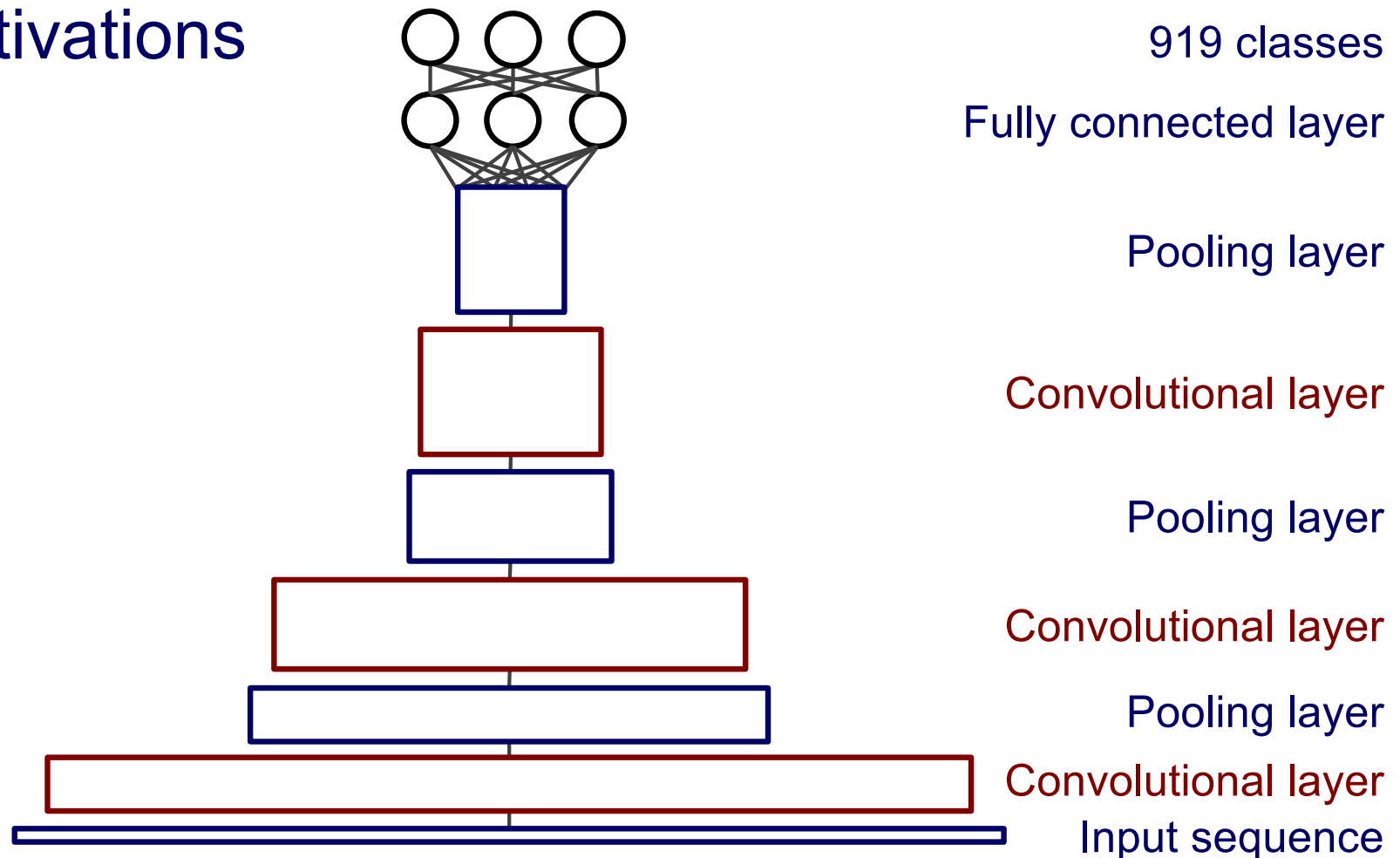


Once again,
shared weight
vector for all
nodes in a row



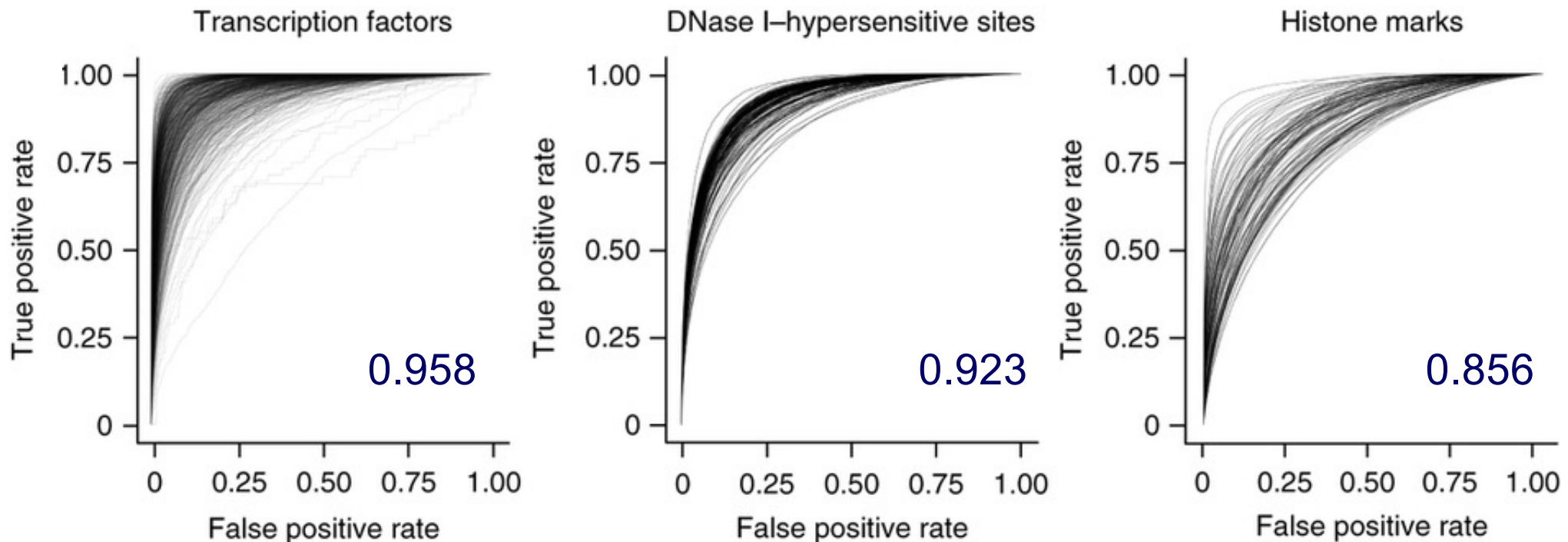
Full DeepSEA neural network

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



Predicting epigenetic annotations

- Compute median AUROC for three types of classes



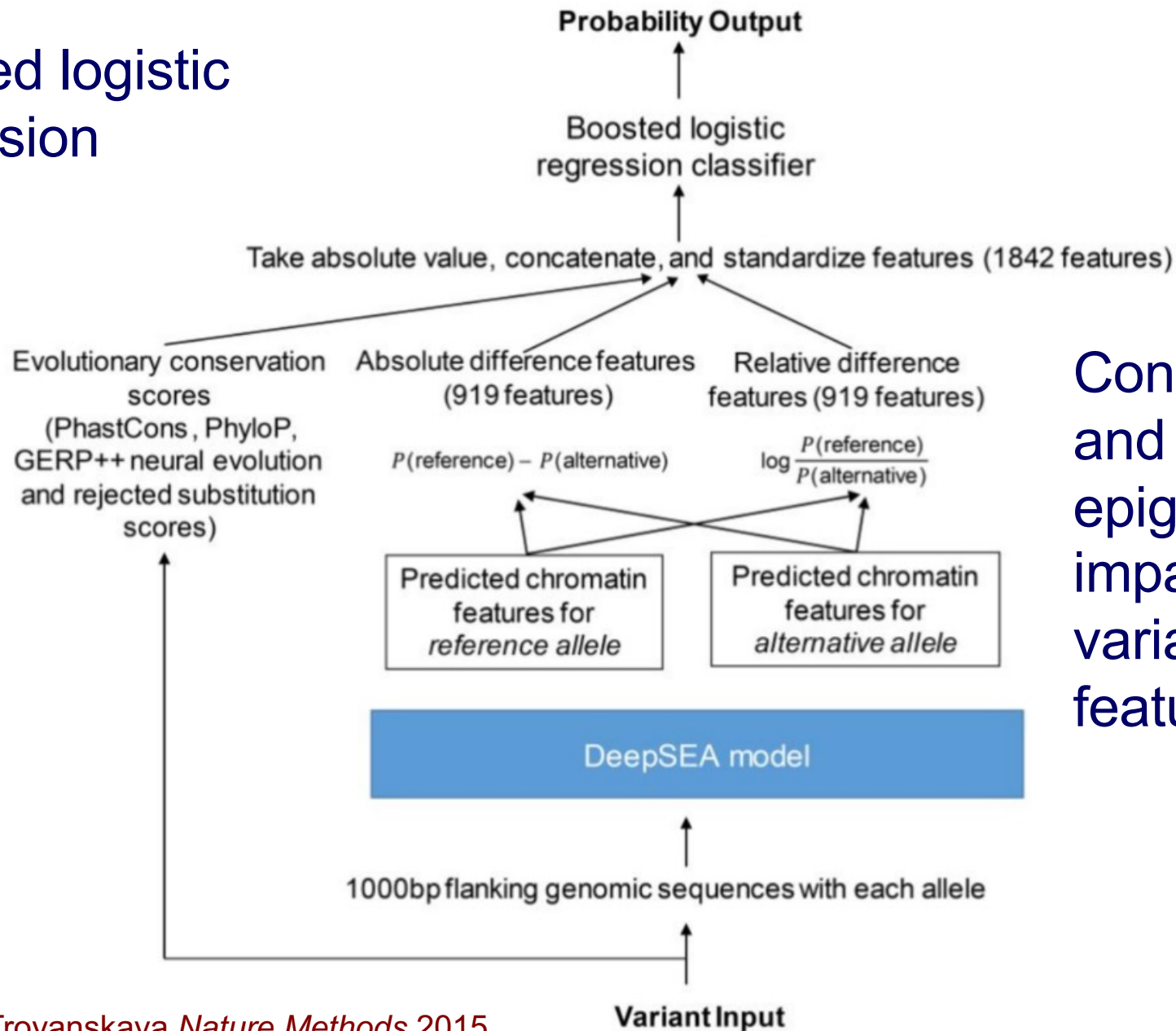
Zhou and Troyanskaya *Nature Methods* 2015

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

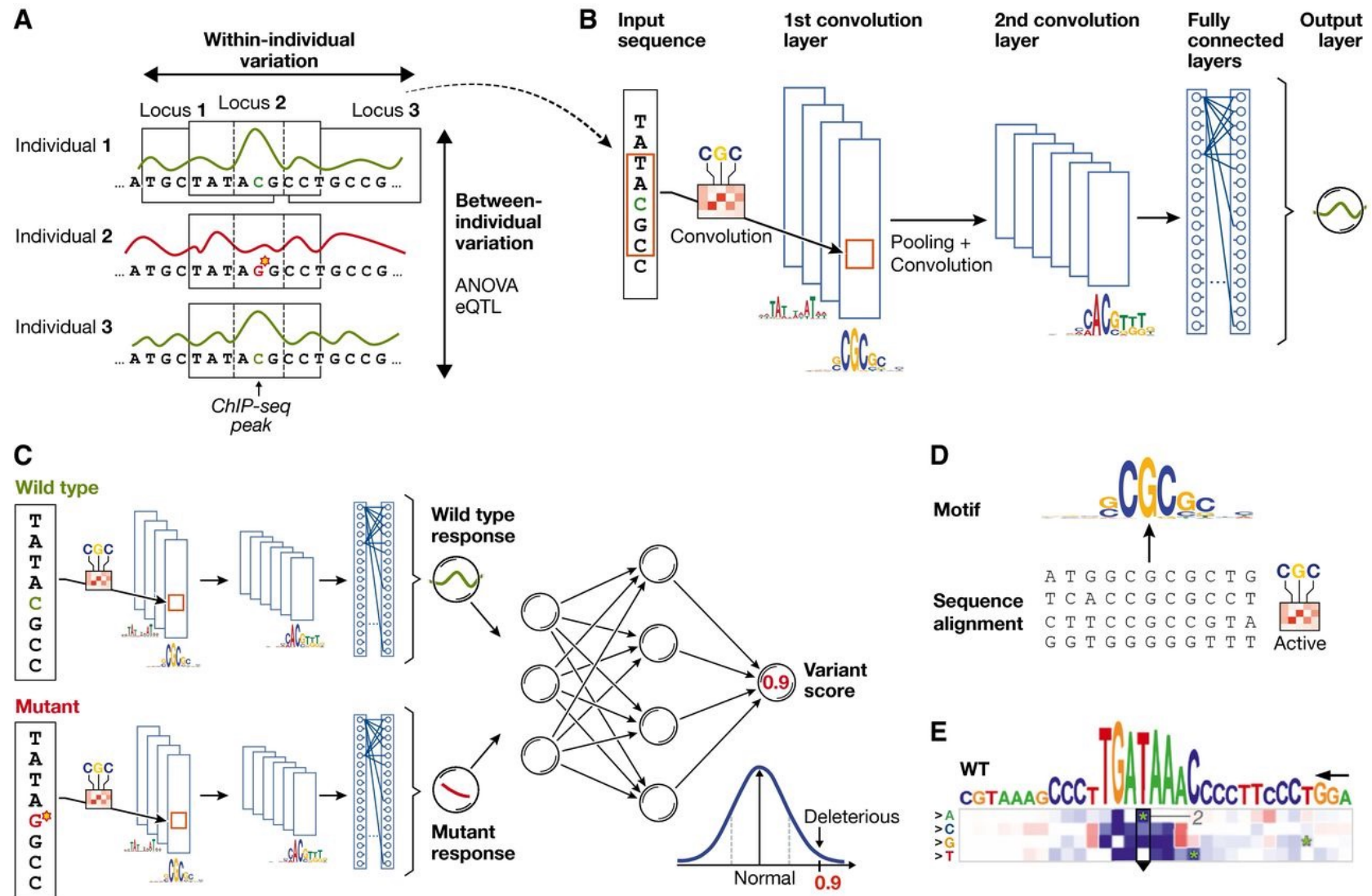
Predicting functional variants

Boosted logistic regression



Conservation and predicted epigenetic impact of variant as features

Principles of using neural networks for predicting molecular traits from DNA sequence



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

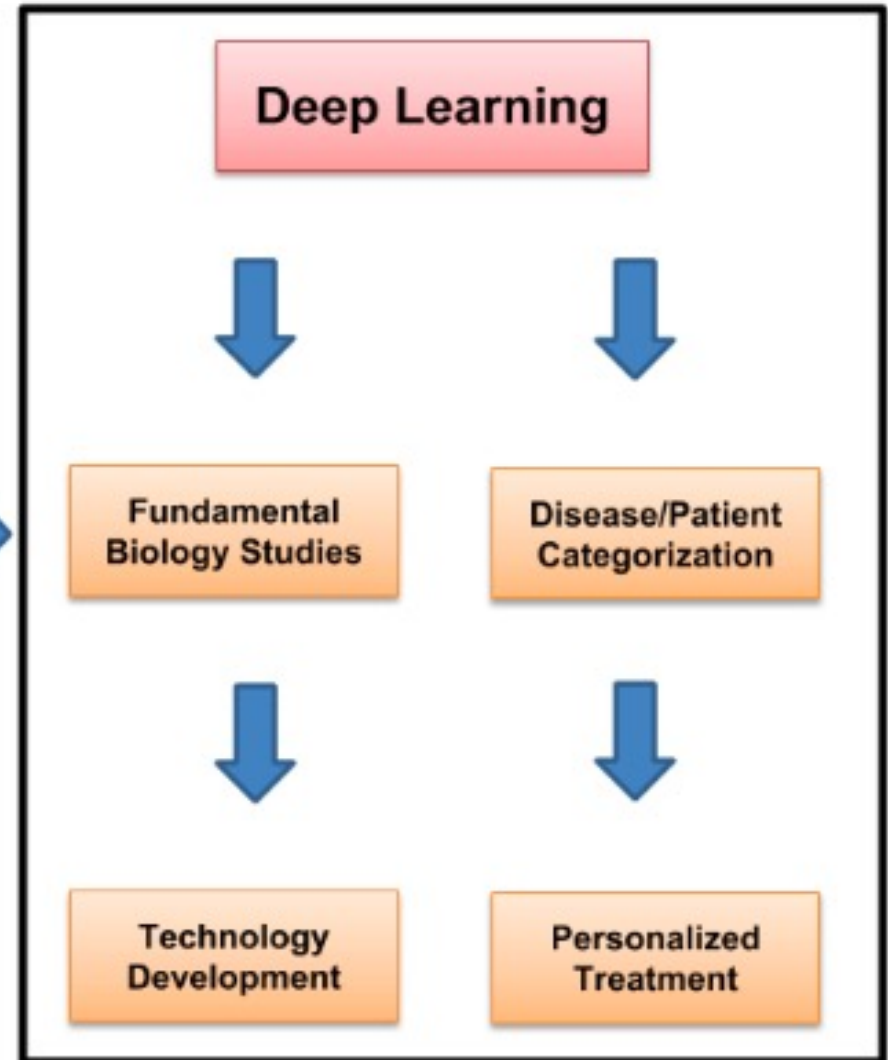
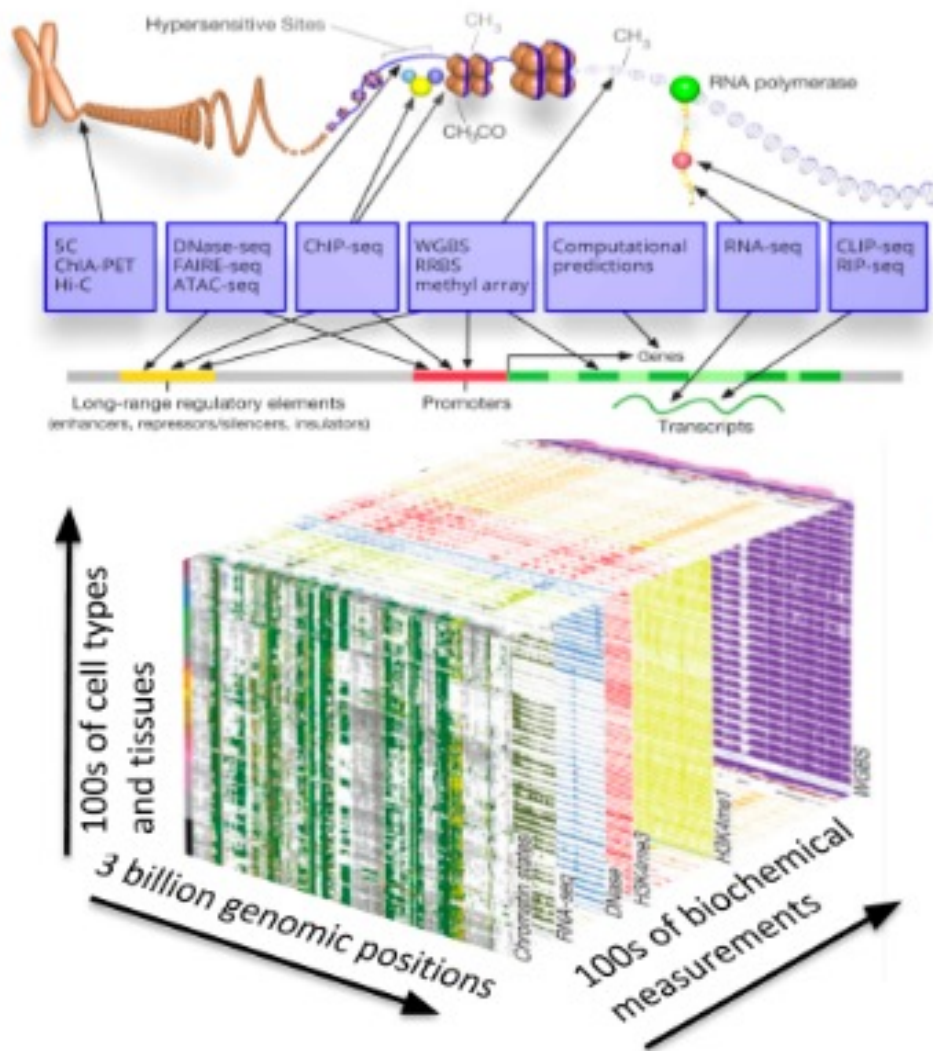
Deep learning is rampant in biology and medicine

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

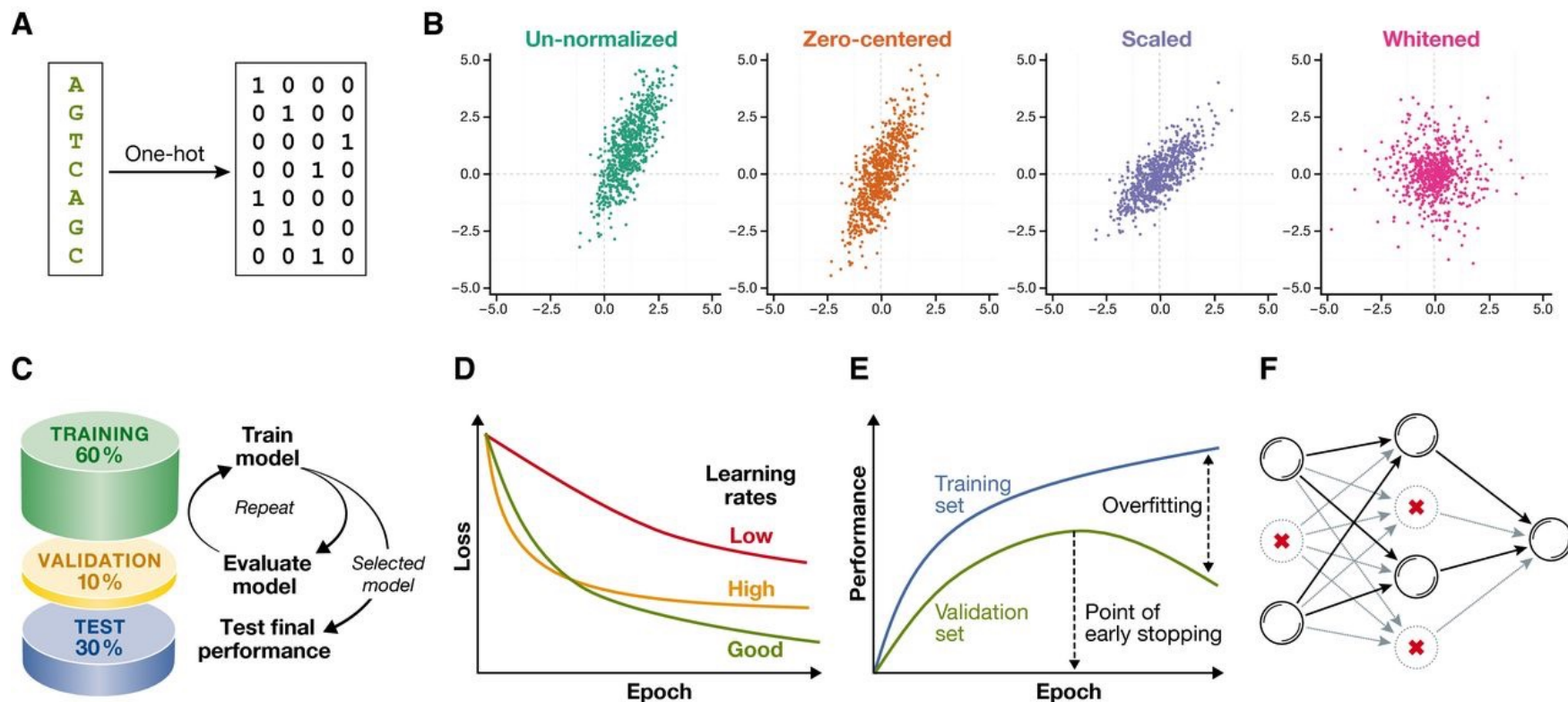
Reading list

- **Deep learning for computational biology**
 - <http://msb.embopress.org/content/12/7/878>
- **Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning**
 - <https://www.nature.com/articles/nbt.3300>
- <https://github.com/hussius/deeplearning-biology>
- [The Incredible Convergence Of Deep Learning And Genomics](#)

The Incredible Convergence Of Deep Learning And Genomics



Data normalization for and pre-processing for deep neural networks



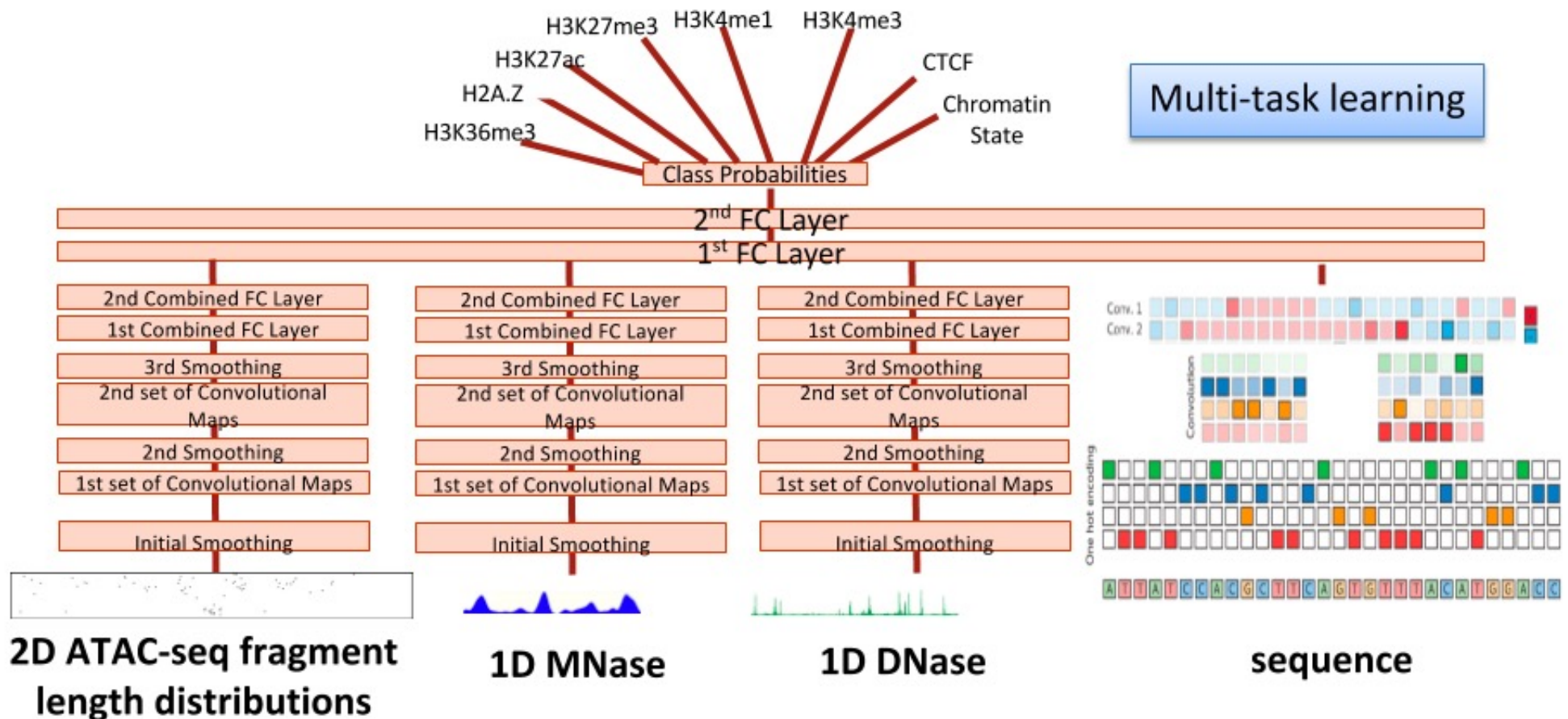
Christof Angermueller et al. Mol Syst Biol 2016;12:878

Overview of existing deep learning frameworks, comparing four widely used software solutions

	Caffe	Theano	Torch7	TensorFlow
Core language	C++	Python, C++	LuaJIT	C++
Interfaces	Python, Matlab	Python	C	Python
Wrappers		Lasagne, Keras, sklearn-theano		Keras, Pretty Tensor, Scikit Flow
Programming paradigm	Imperative	Declarative	Imperative	Declarative
Well suited for	CNNs, Reusing existing models, Computer vision	Custom models, RNNs	Custom models, CNNs, Reusing existing models	Custom models, Parallelization, RNNs

THE CHROMPUTER

Integrating 1D, 2D signals, and sequence to predict multiple outputs



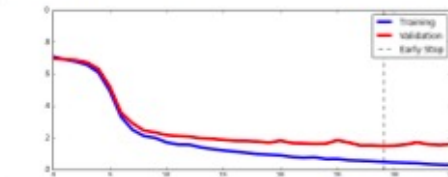
How to train your DragoNN

- https://drive.google.com/file/d/0B4Yo77Kh_QeeaXZKQUtZWjNrWkE

```
motif_density_localization_simulation_parameters = {  
    "motif_name": "TAL1_known4",  
    "seq_length": 1000,  
    "center_size": 150,  
    "min_motif_counts": 2,  
    "max_motif_counts": 4,  
    "num_pos": 3000,  
    "num_neg": 3000,  
    "GC_fraction": 0.4}
```

```
one_filter_dragonn_parameters = {  
    'seq_length': 1000,  
    'num_filters': [1],  
    'conv_width': [10],  
    'pool_width': 35}
```

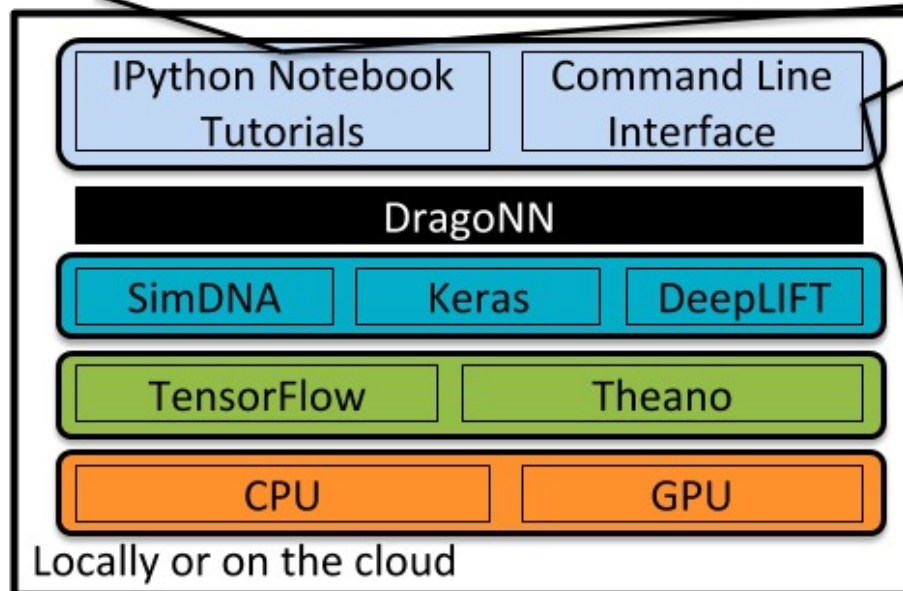
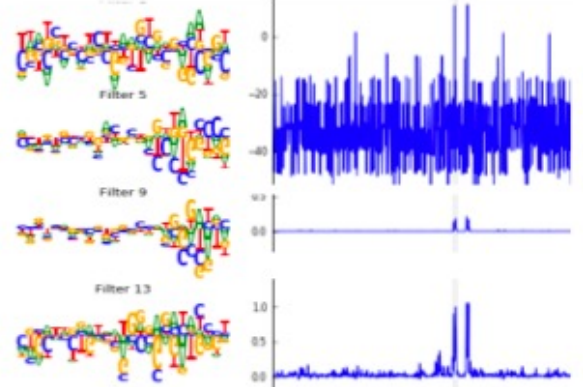
SequenceDNN_learning_curve(one_filter_dragonn)



interpret_SequenceDNN_filters(multi_layer_dragonn, simulation_data)



interpret_data_with_SequenceDNN(multi_filter_dragonn, simulation_data)



usage: dragonn [-h] {train,test,predict,interpret}

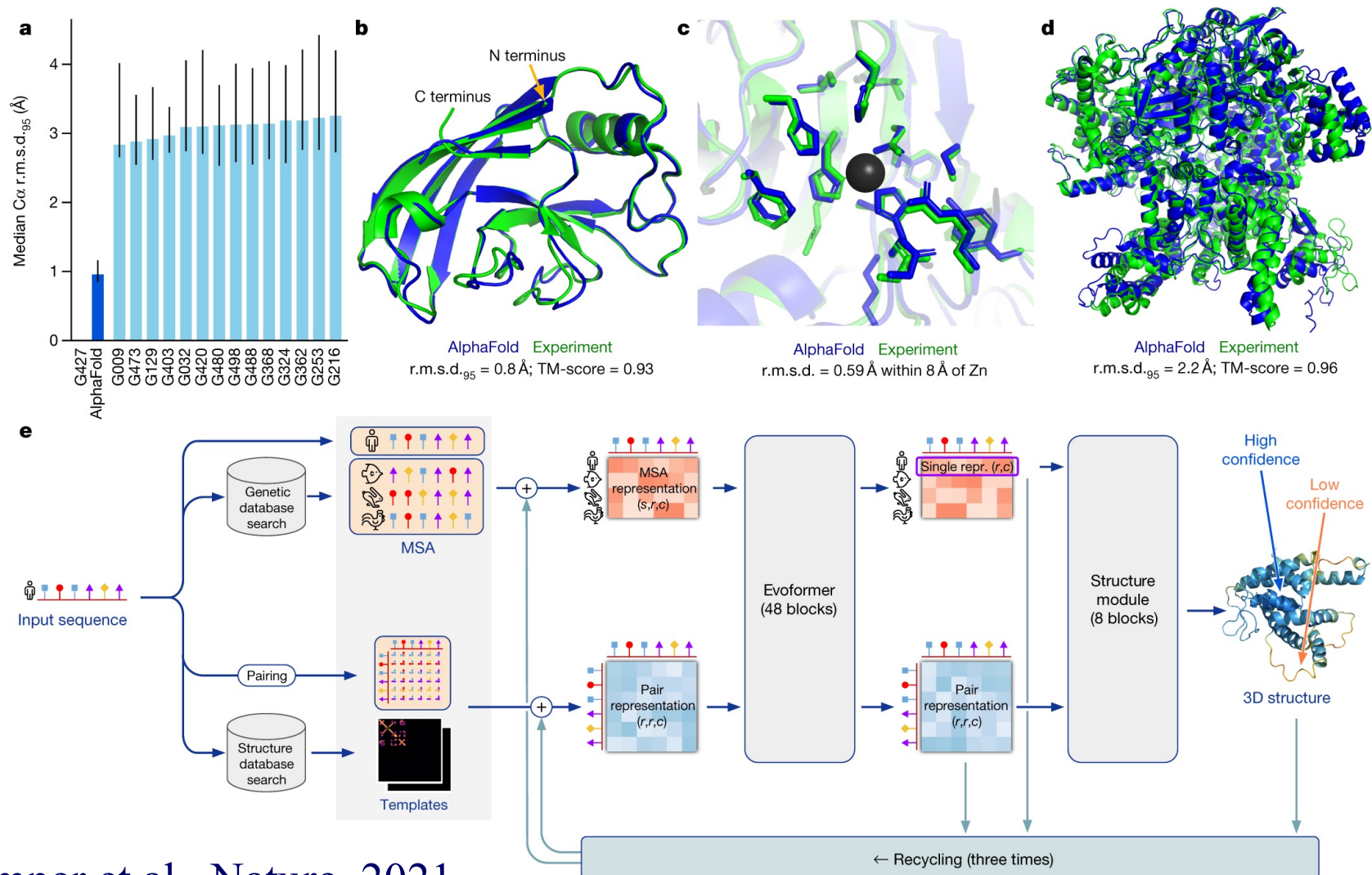
main script for DragoNN modeling of sequence data.

positional arguments:

{train,test,predict,interpret}

	dragonn command help
train	model training help
test	model testing help
predict	model prediction help
interpret	model interpretation help

AlphaFold predicts accurate protein structures



Jumper et al., Nature, 2021