# Recitation 5: RNA-seq, splicing, additional machine learning topics

6.784: DEEP LEARNING IN THE LIFE SCIENCES

# Deep convolutional neural network

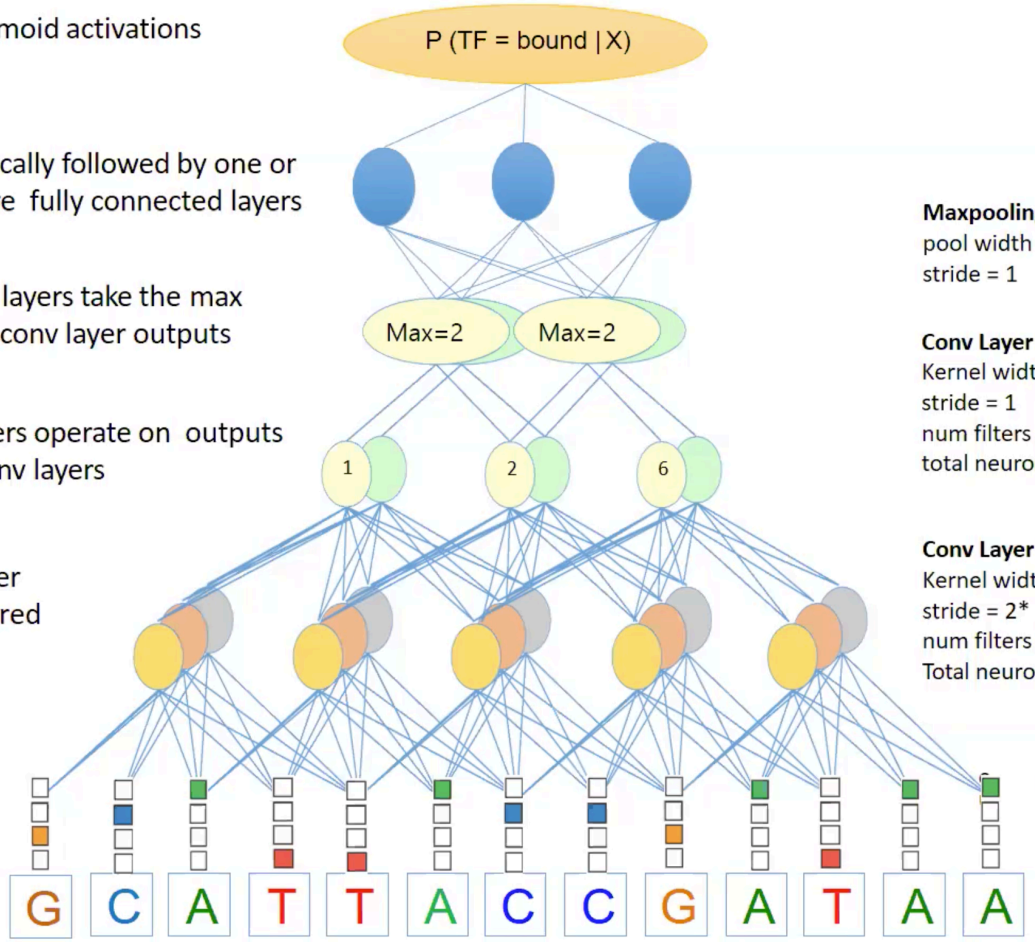

Sigmoid activations

Typically followed by one or more fully connected layers

Maxpooling layers take the max over sets of conv layer outputs

Later conv layers operate on outputs of previous conv layers

Convolutional layer (same color = shared weights)



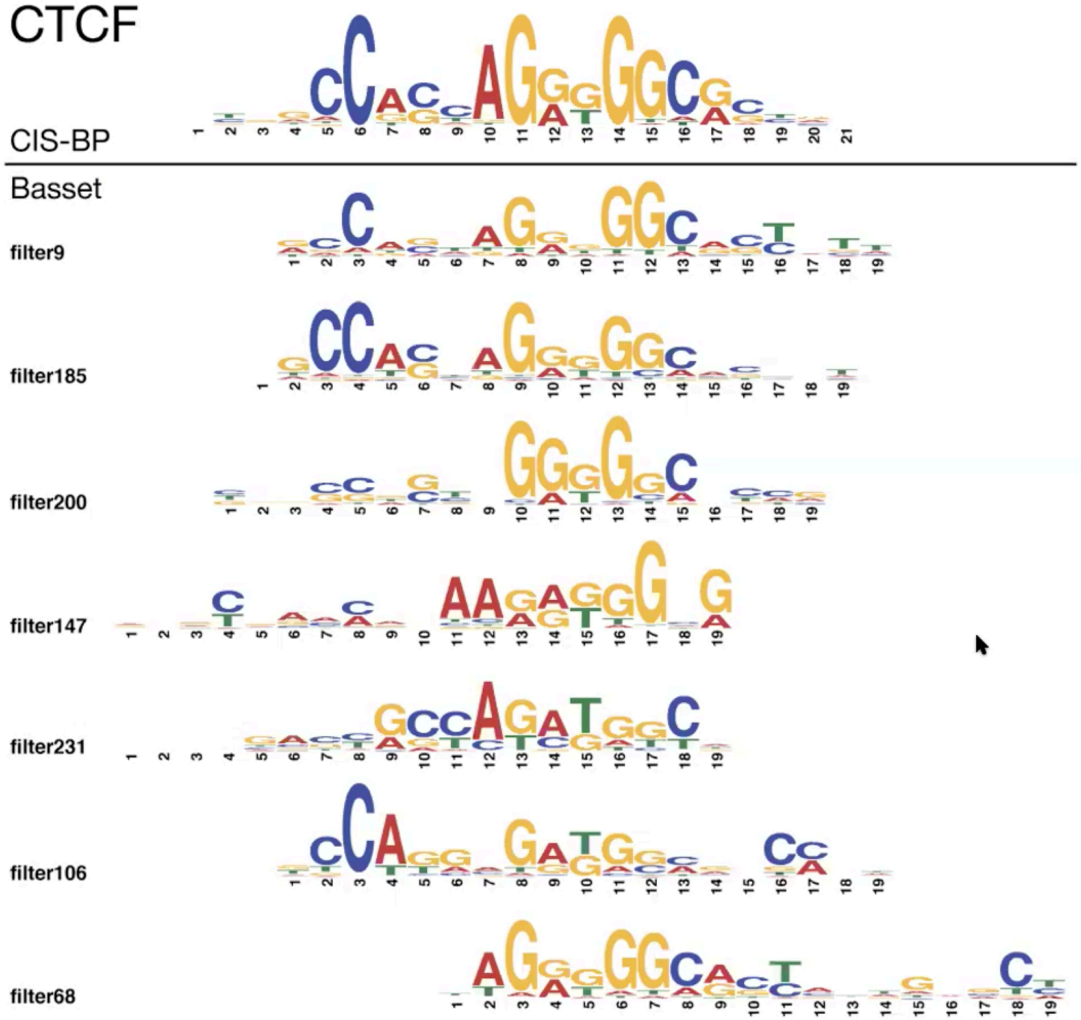

**Maxpooling layer**
pool width = 2
stride = 1

**Conv Layer 2**
Kernel width = 3
stride = 1
num filters / num channels = 2
total neurons = 6

**Conv Layer 1**
Kernel width = 4
stride = 2*
num filters / num channels = 3
Total neurons = 15

*for genomics, a stride of 1 for conv layers is recommended

# Multiple filters capture motif variants

Inputs and Outputs

1. Inputs are DNA sequences with one-hot encoded nucleic acids (4 channels, [A, T, C, G]).
   - shape `(?, 1, 101, 4)`
2. Labels are one-hot encoded markers of the classes [bound, unbound].
   - shape `(?, 2)`
3. Outputs are softmax probabilities for the classes [bound, unbound].
   - shape `(?, 2)`

| Class Name | For model layers (see model spec): |
|---|---|
| `ConvLayer` | layer 1 |
| `MaxPoolLayer` | layer 2 |
| `FCLayer` | layers 3 & 5 |
| `DropoutLayer` | layer 4 |

## `DropoutLayer` class definition

Formulae:

$$y_i = x_i d_i,$$

where

- $i \in \{1, 2, \ldots, N\}$
- $d_i \in \{0, 1\}$
- $P(d_i = 0) = r$
- $N$: number of elements in $x$
- $r$: dropout rate

## `ConvLayer` class definition

Formulae:

$$y_k = \phi\left((x * w_k) + b_k\right),$$

where

- $k \in \{1, 2, \ldots, K\}$
- $K$ is the number of kernels
- $*$ is the discrete convolutional operator
- $b_k$ is the scalar bias for each kernel
- $\phi$ is the activation function

## `MaxPoolLayer` class definition

Formulae:

$$y_{i',\,j'} = \max \begin{bmatrix} x_{i,\,j} & \cdots & x_{i,\,j+(s-1)} \\ \vdots & \ddots & \\ x_{i+(s-1),\,j} & & x_{i+(s-1),\,j+(s-1)} \end{bmatrix},$$

where

- $i' \in \left\{1, 2, \ldots, \lceil \frac{M}{s} \rceil\right\}$
- $j' \in \left\{1, 2, \ldots, \lceil \frac{N}{s} \rceil\right\}$
- $i = i'\,(s-1) + 1$
- $j = j'\,(s-1) + 1$
- $s$: pooling size
- $M$: input length
- $N$: input width
- $x_i|_{i>M} = x_j|_{j>N} = 0$ (i.e. zero padding)

- Model Architecture, Sequential Network Layers
  1. Convolutional layer with
     - 32 kernels
     - kernel length: 11
     - convolution stride: 1
     - ReLU activation
     - padding: `same`, such that the (input shape) = (output shape)
  2. Max-pooling layer with
     - pooling size: 2
     - pooling stride: 2
     - padding: `same`
  3. Fully connected layer with
     - 64 neurons (outputs)
     - ReLU activation
     - *Note: to perform the linear computation, you will need to flatten the input into a 2D tensor if the input has* `tf.rank` $> 2$.
  4. Dropout layer with
     - dropout rate: $[1 - 10^{-3}, 0.8, 0.5, 0.2]$
       *Note:* (dropout rate) $= 1 -$ (keep probability)
  5. Fully connected layer with
     - 2 neurons (outputs)
     - softmax activation

## Problem 5B: ROC Curve

You'll need to define the following function for computing the receiver operating characteristic (ROC) curve and the area under the ROC curve. The two axes of the curve are $\mathrm{FPR}$: $x$-axis and $\mathrm{TPR}$: $y$-axis. Please see the confusion matrix Wikipedia page (https://en.wikipedia.org/wiki/Confusion_matrix) for more information. Remember that we're computing these statistics for a **binary classifier** (bound vs. unbound).

Useful formulas:

$$\begin{aligned}
\mathrm{P} &: \text{real number of positives in the data} \\
\mathrm{N} &: \text{real number of negatives in the data} \\
\mathrm{TP} &: \text{true positives} \\
\mathrm{FP} &: \text{false positives} \\
\mathrm{TPR} &: \text{true positive rate} = \frac{\mathrm{TP}}{\mathrm{P}} \\
\mathrm{FPR} &: \text{false positive rate} = \frac{\mathrm{FP}}{\mathrm{N}}
\end{aligned}$$

$$||\mathbf{w}||_1 = |w_1| + |w_2| + ... + |w_N|$$

1-norm (also known as L1 norm)

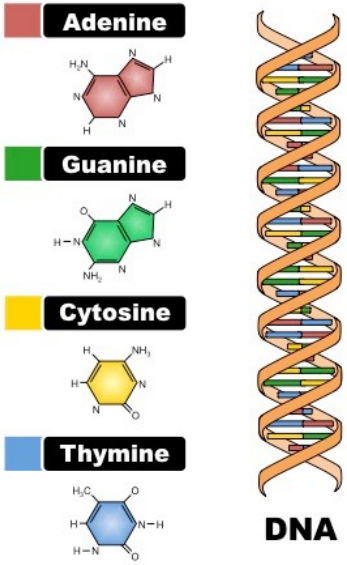
$$\|\mathbf{w}\|_2 = \left(|w_1|^2 + |w_2|^2 + \ldots + |w_N|^2\right)^{\frac{1}{2}}$$

2-norm (also known as L2 norm or Euclidean norm)

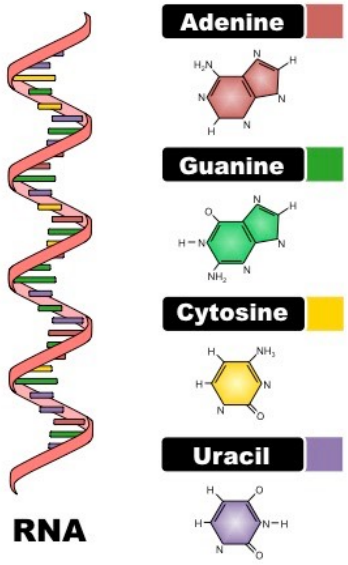
$$\|\mathbf{w}\|_p = \left(|w_1|^p + |w_2|^p + \ldots + |w_N|^p\right)^{\frac{1}{p}}$$

p-norm

# Nucleic Acids




**DNA**
(Deoxyribonucleic Acid)



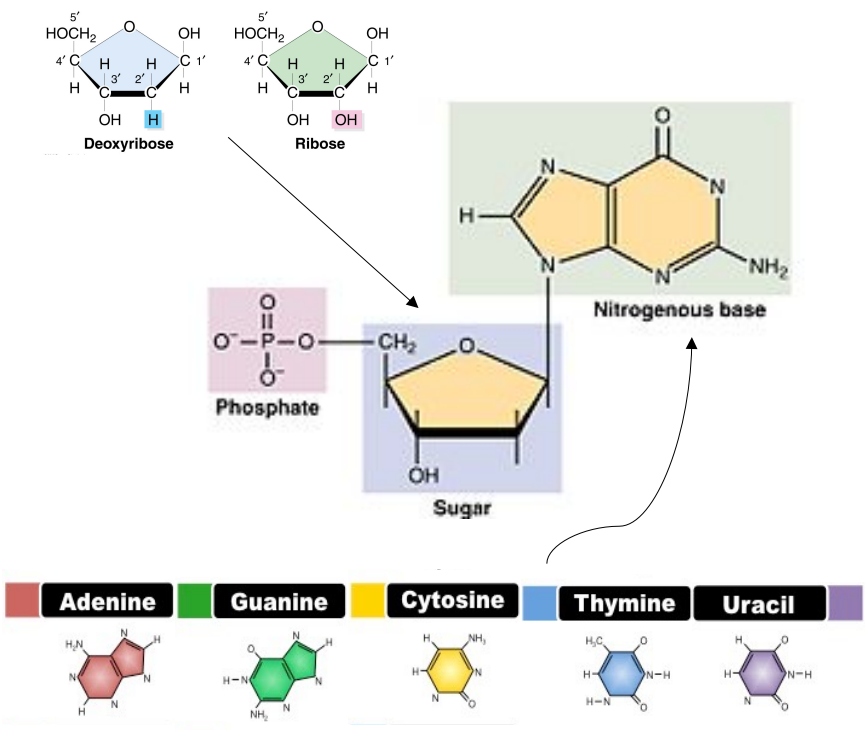

**RNA**
(Ribonucleic Acid)

# Nucleic Acids

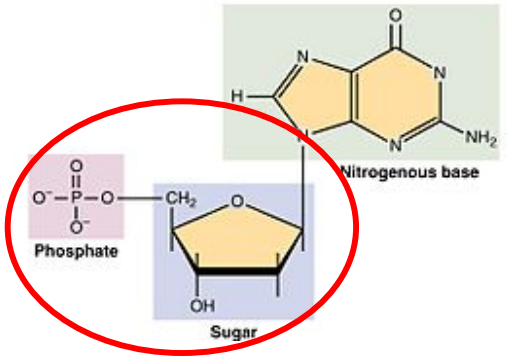
- **Nucleic acids (DNA and RNA) are composed of "units" called nucleotides**
  - Nucleotides are the "building blocks" of nucleic acids (DNA and RNA)
  - 5 different nucleotides (differentiate them based on their **nitrogenous base**: A, G, C, T, U)

- **Nucleotide "building blocks" have 3 parts:**
  - **Phosphate**
  - **Sugar:** ribose in RNA, deoxyribose in DNA
  - **Base:** Adenine (A), Guanine (G), Cytosine (C), and Thymidine (T) (DNA) or Uracil (U) (RNA)
  - Different nucleotides are defined by their base (A, G, C, T, U)

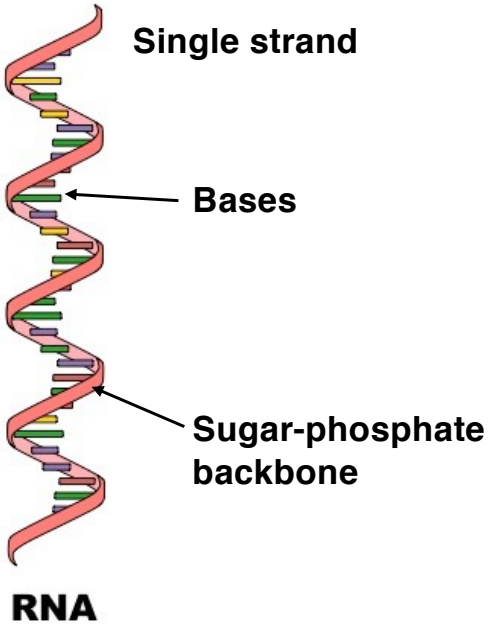
- **Function: Nucleic acids (DNA and RNA) carry information about making proteins**
  - Information is coded by the nucleotide bases A, C, G, and T (DNA) or U (RNA)
  - The order or sequence of the nucleotides provides information



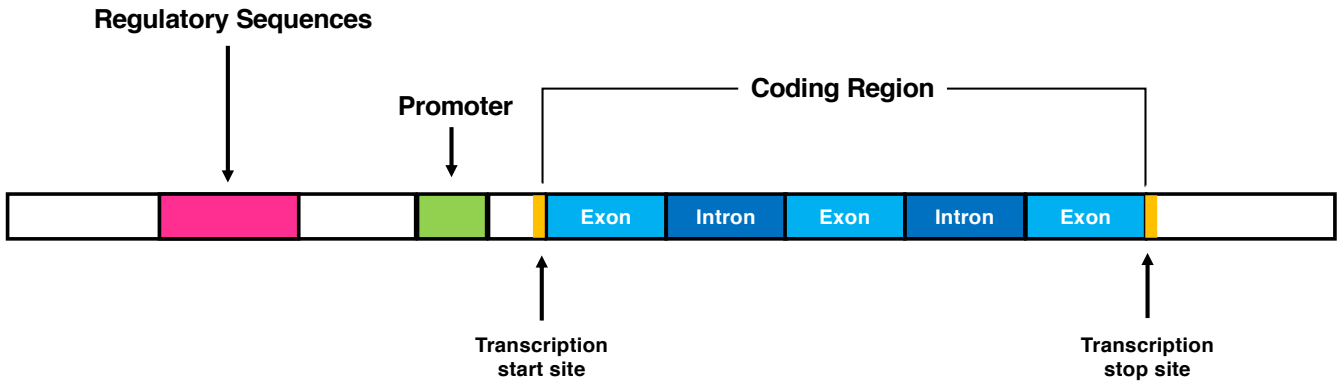
# RNA: breaking down the structure




- Nucleotides with different bases (A, G, C, U) link together via sugar/phosphate regions to form a RNA strand



# Three important parts of a gene




**Regulatory Sequences:** Don't worry about this (for now)

**(1) Promoter:** Can affect rate and frequency at which protein is made and can also turn gene "on" and "off"

**(2) Coding region:** Contains actual information (sequence) to make the protein and is made up of exons and introns (introns are removed from RNA via "splicing" but are present in the DNA)

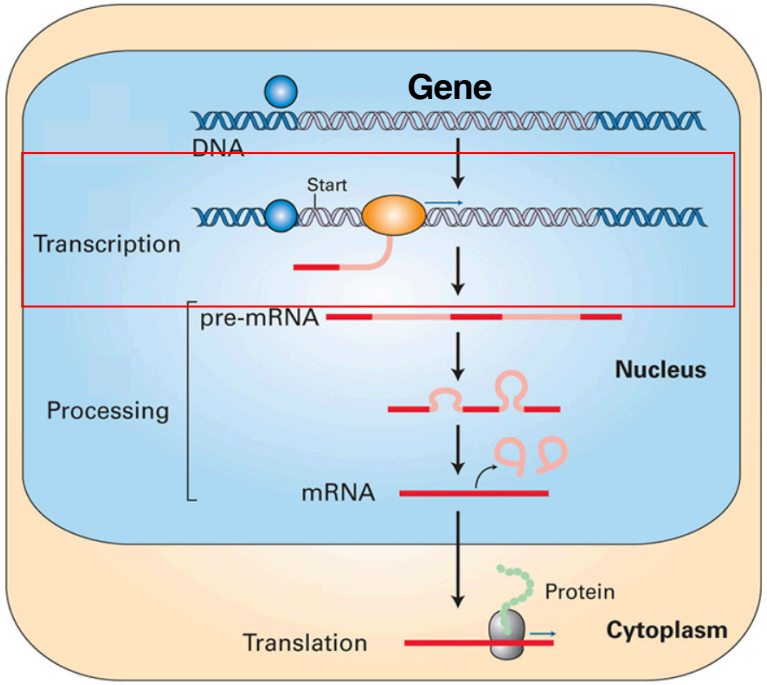
**Exon:** contain information coding for a protein
**Intron:** does not contain protein information

**(3) Transcription start/stop sites:** Where transcription starts and ends respectively

# Protein production step 1: transcription (DNA to RNA)

**Transcription goal:** convert genetic information in DNA to RNA (mRNA form)

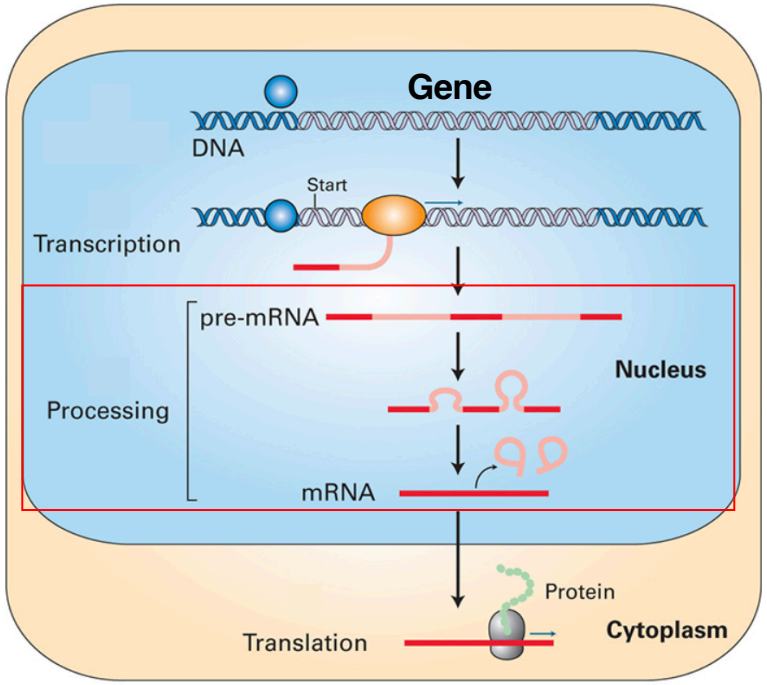
- **DNA in gene coding for protein is "transcribed"**
  - Transcription: protein coding region of a gene is copied into mRNA
  - mRNA sequence = same as one of DNA strands
- DNA contains introns and exons thus, **pre-mRNA** also contains introns and exons



# Protein production step 1: transcription (DNA to RNA)

**Transcription goal:** convert genetic information in DNA to RNA (mRNA form)

- **DNA in gene coding for protein is "transcribed"**
  - Transcription: protein coding region of a gene is copied into mRNA
  - mRNA sequence = same as one of DNA strands
- DNA contains introns and exons thus, **pre-mRNA** also contains introns and exons
- **Pre-mRNA processing**
  - Introns get cut out and exons are joined together
  - Term for this is "splicing"
  - Other modifications also made
- **Mature mRNA is ready for the next step:**
  - Translation (RNA→ protein)



# Transcription: a closer look

**RNA Polymerase:** enzyme that synthesizes mRNA molecule using the DNA strand as a template

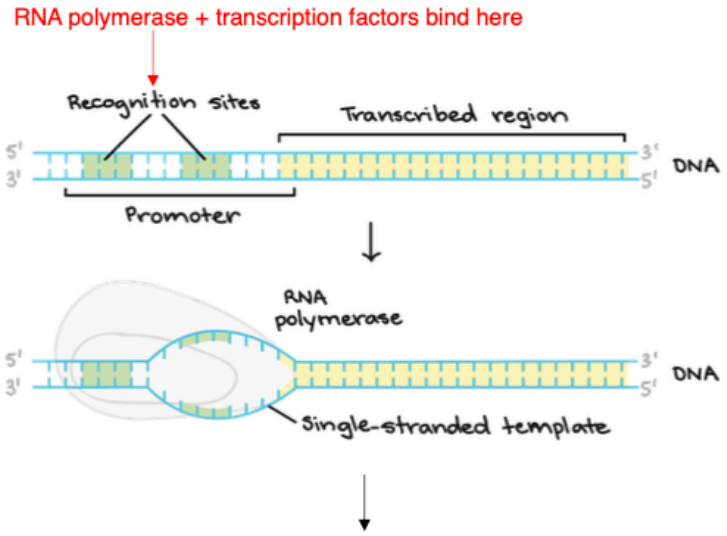
**Transcription factor:** protein that binds to specific DNA sequence and controls rate of transcription – in general they function to regulate genes (turn them on/off)

**Three big steps:**

**(1) Initiation:**
- RNA polymerase + 1 or more transcription factors bind to promoter region
- RNA polymerase separates 2 strands of the DNA helix ("transcription bubble")

**(2) Elongation**
- RNA polymerase begins transcription at start site
- RNA polymerase adds RNA nucleotides to mRNA strand (copied off of DNA strand)

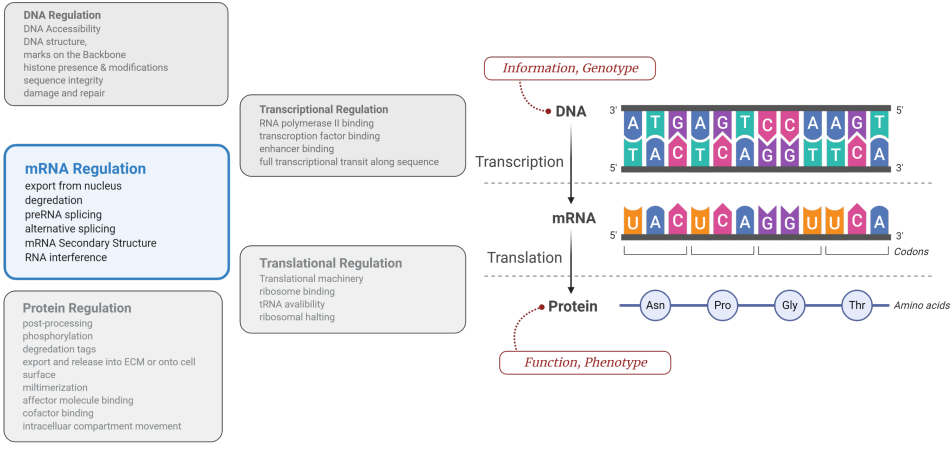
**(3) Termination**
- RNA polymerase releases from template DNA
- Complete pre-mRNA molecule dissociates




**Pre-mRNA molecule**

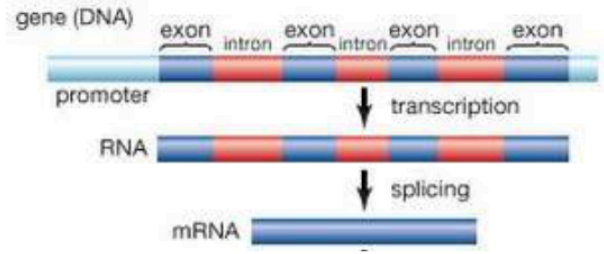
# Gene Splicing Overview I

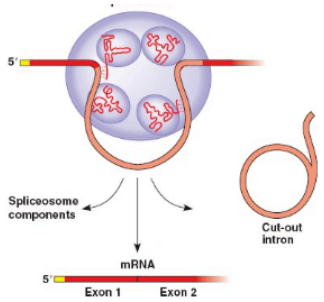
RNA Splicing as an element of the regulatory mechanisms that affect the mRNA molecule, the second key molecule of the central dogma.



# Splicing: removing introns from pre-mRNA

- **Splicing:** removing introns from a pre-mRNA transcript to form a mature mRNA transcript
- Exons contain the actual protein coding information
- Exons are made up of codons which can then be read by the ribosome
- Introns do not contain coding information
- **Spliceosome:** the machinery that actually removes the introns (can recognize characteristic "splice sites" between introns and exons)

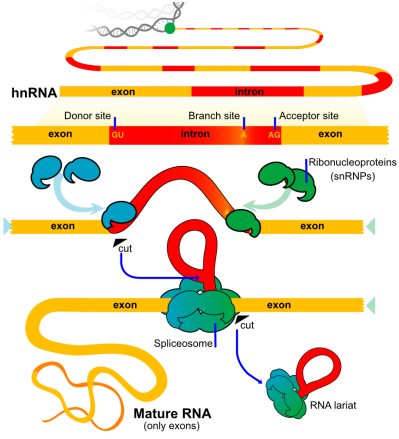



**Spliceosome:** large molecular "machine" comprised of proteins and small nuclear RNAs

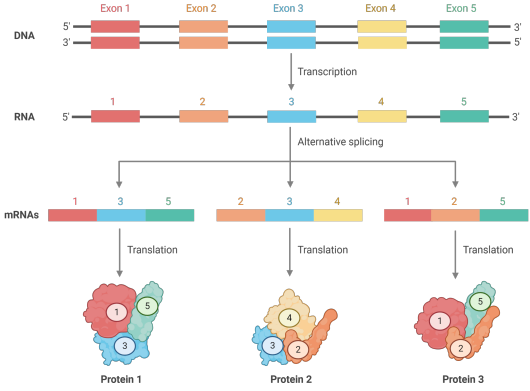
# Gene Splicing Overview II

For splicing to occur (1) a number of protein components known as the "splicosome" must bind to the mRNA and (2) the prePRNA must adopt a specific secondary structure which brings the 3' end of the first exon in proximity to the 5' end of the following exon.



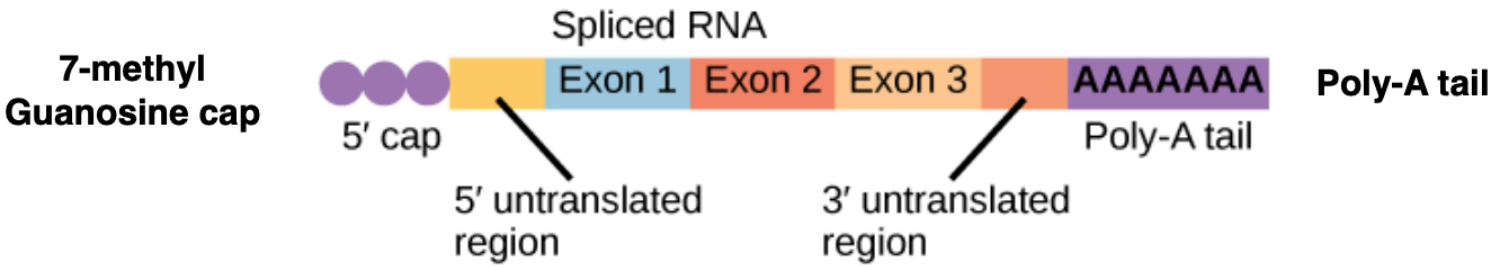
# Gene Splicing Overview IV

Furthermore, these regulatory elements make it possible for a single preRNA transcript to yeild many different mRNA products and therefore different protein products.




Dysregulation of these alternative splicing outcomes is contributor protein dysfunction and trascriptiome instability, contributing to pathologies including cancer and drug addiction.

# Other Modifications



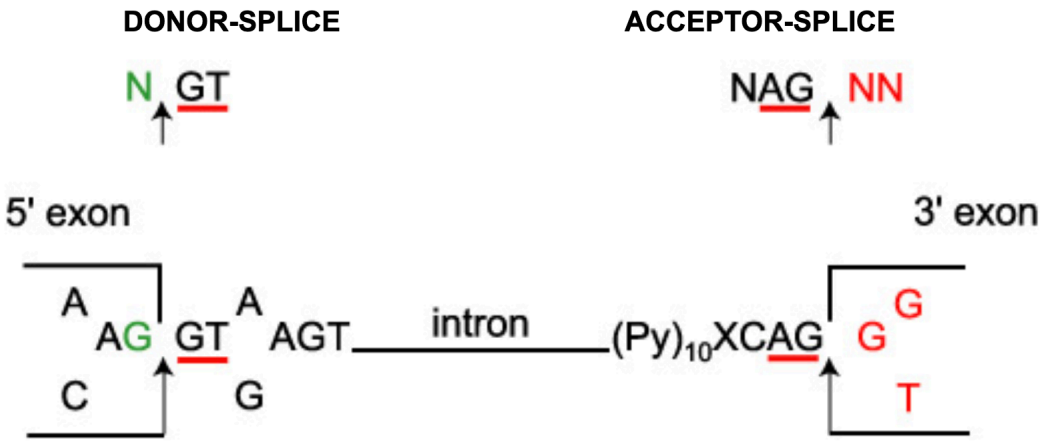

**These end modifications:**

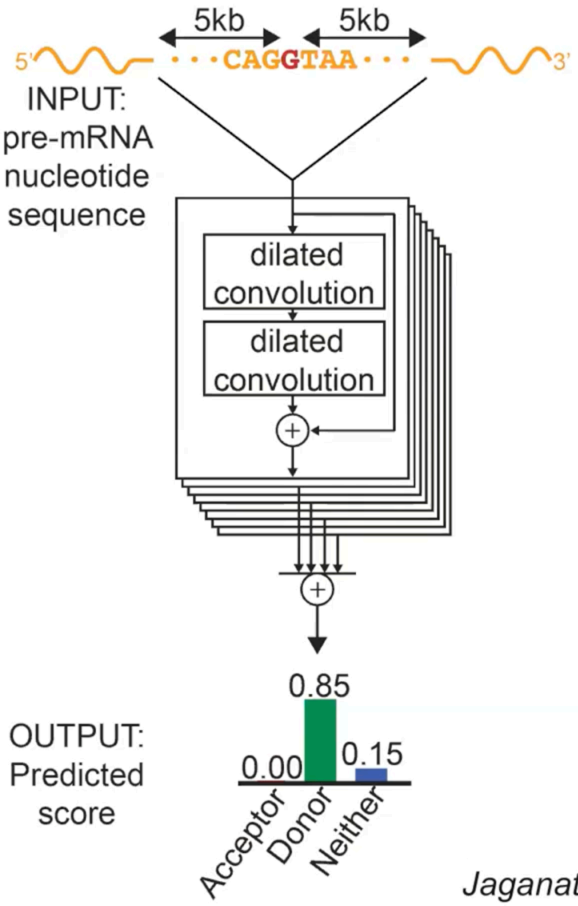
- Increase mRNA stability (prevent degradation by enzymes)
- Assist with mRNA transport out of nucleus
- Promote translation (mRNA → ribosome)

# Splicing sites

**IMGT labels:**

Colored letters in that figure correspond to splicing frame 1




# SpliceAI

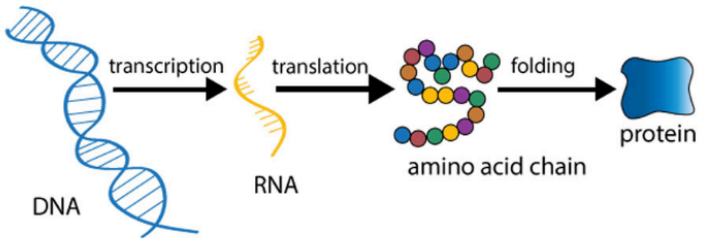
- **Input**: 10K nucleotides
- **Labels**: 3-way classification, based on GENCODE annotations & RNA-seq
- **Architecture**: 32-layer convolutional neural network, 700K parameters
- Trained on half of chromosomes, withheld other half for testing, excluding paralogs

*Jaganathan et al, Cell 2019*

# Gene Expression

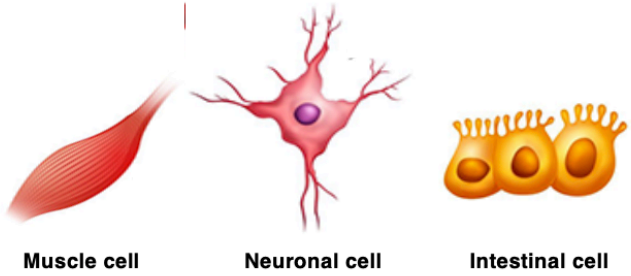
**What does it mean to "express" a gene?**

- Expression just means that whatever that gene codes for is being made (the gene product is being produced)
  - DNA sequence → mRNA → protein

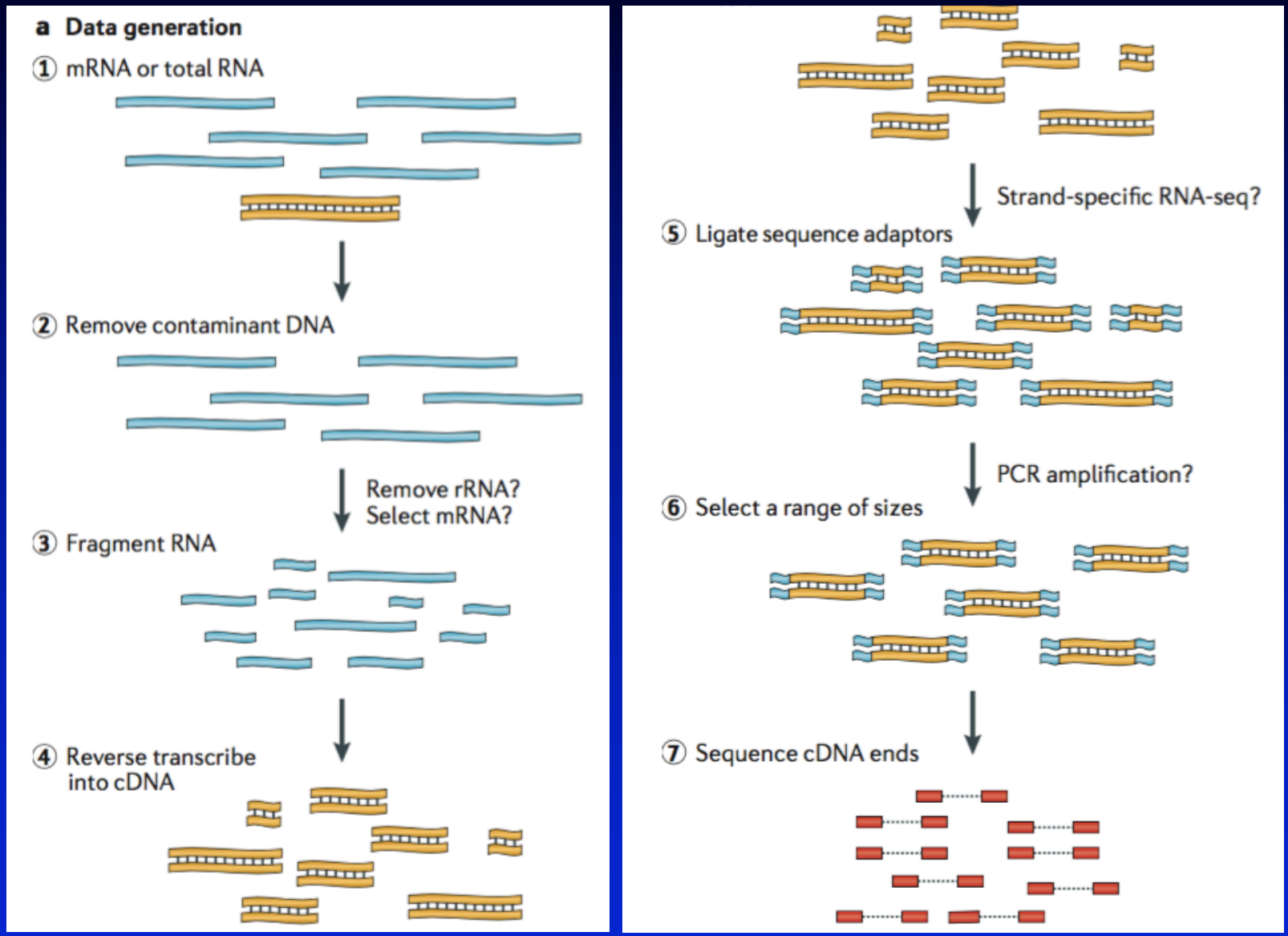



**Why is gene expression important?**

- The genes that a cell expresses dictate what proteins that cell will make
- The proteins a given cell makes determine the cell's identity and functionality
  - (i.e. whether it's a muscle cell, a neuron, a liver cell, an intestinal cell, etc.)



# RNA-seq Protocol






Martin and Wang Nat. Rev. Genet. (2011)

# Why RNA-seq, not Microarray?

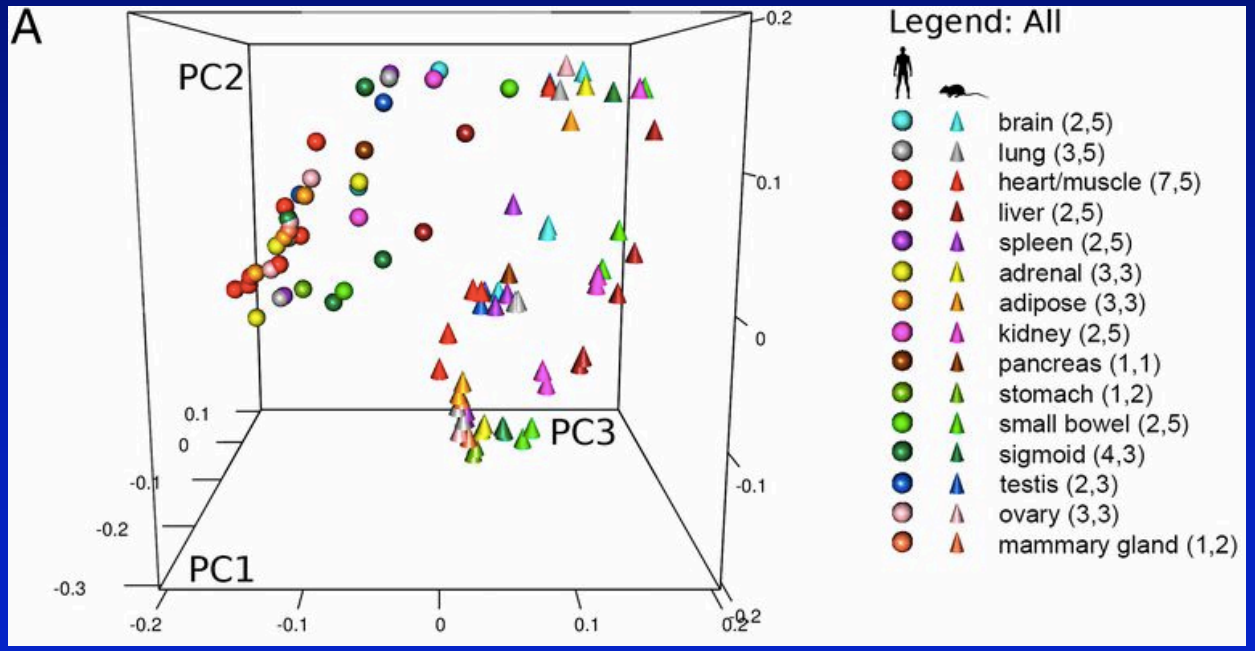
- No need to know the genome sequence or predict genes
- No need to design microarray probes
- Digital representation
- Higher detection range
- New genes
- Alternative splicing
- Mutations and gene fusion

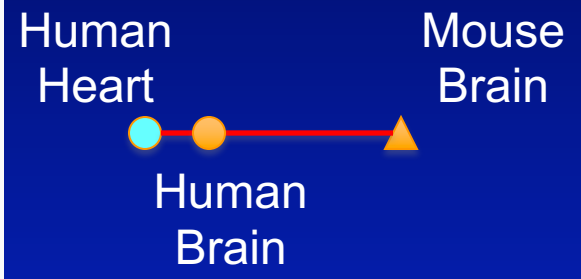
# Experimental Design

- Assessing biological variation requires biological replicates (no need for technical replicates)
- 3 preferred, 2 OK, 1 only for exploratory assays (not good for publications)
- Batch effects still exist, try to be consistent or process all samples at the same time
- Better technology never eliminate the needs for good experimental design

# Batch effect

- Striking finding in 2014: “Human heart is more similar with human brain than mouse brain”?

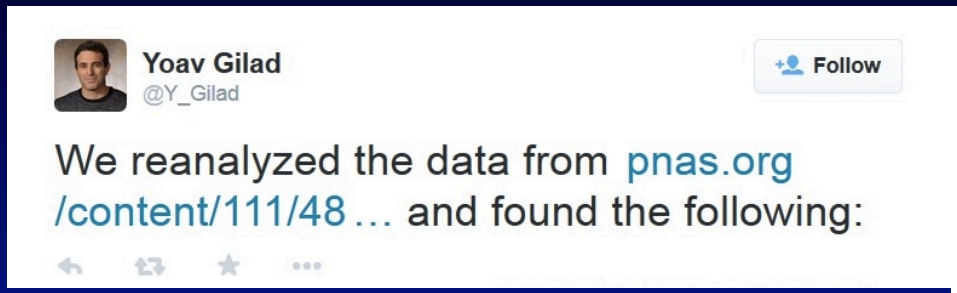


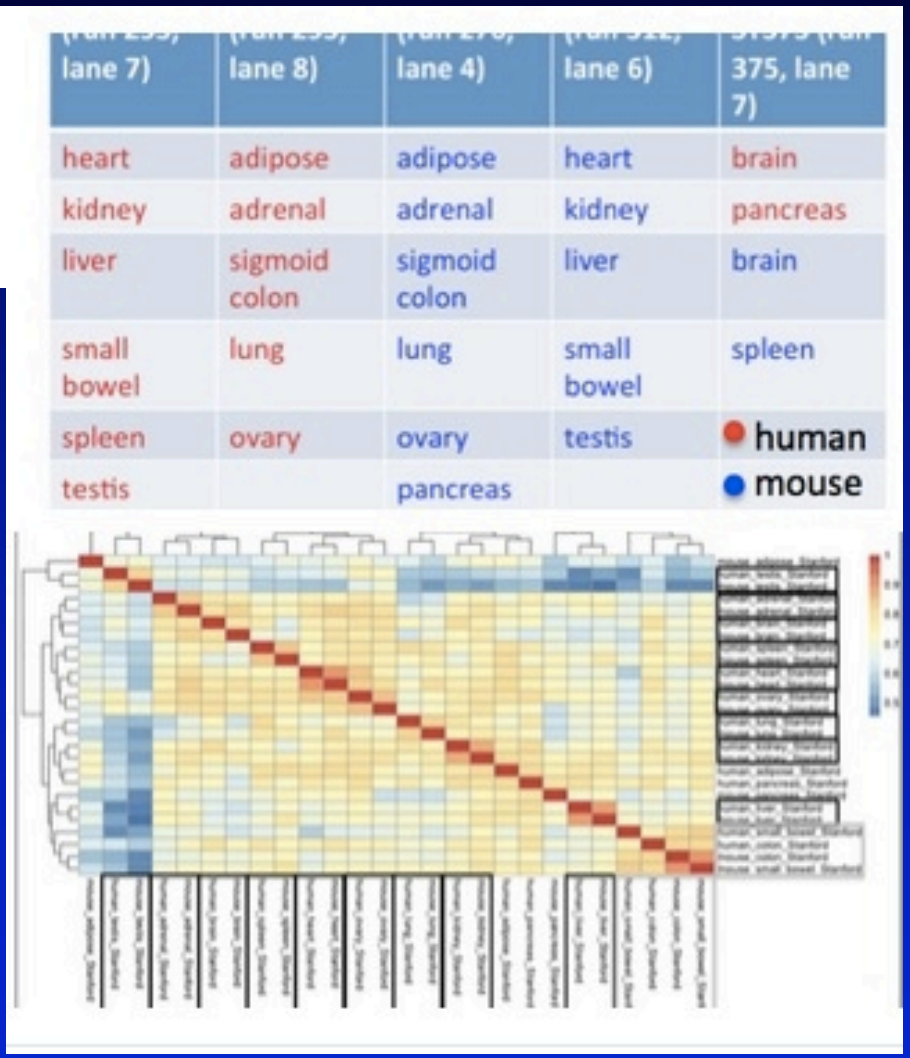



Lin et al, PNAS 2014

# Batch Effect



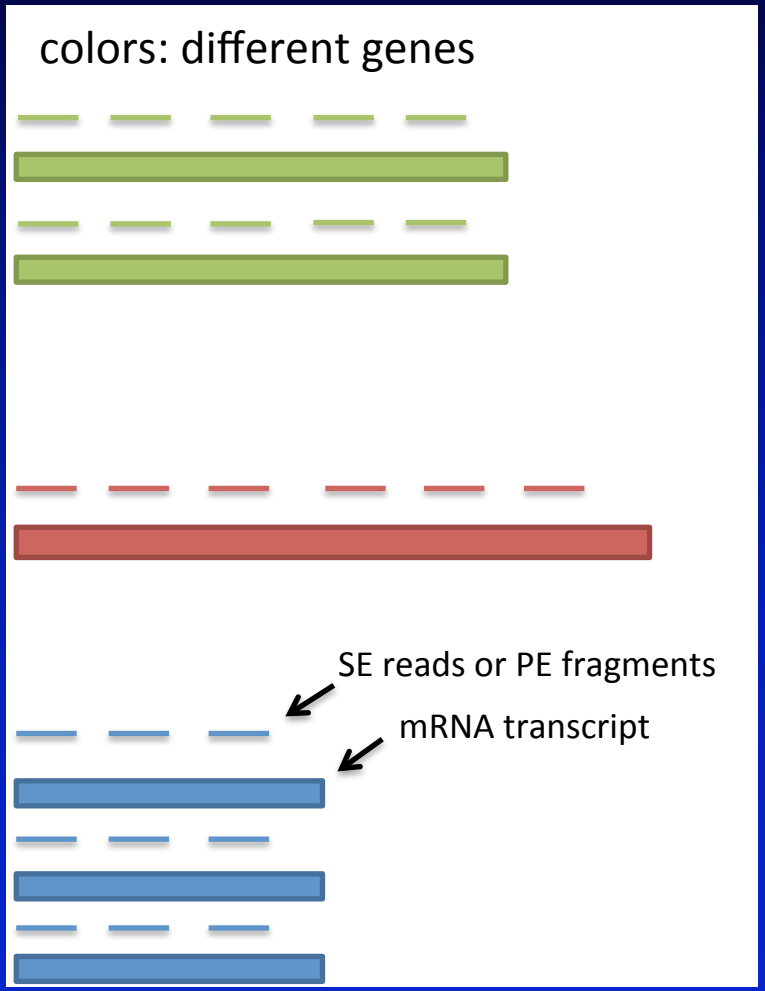

- 1st batch: human tissues
- 2nd batch: human tissues
- 3rd batch: mouse tissues
- 4th batch: mouse tissues
- 5th batch: human/mouse tissues
- After batch removal, tissues cluster




Break

# RNA-seq Abundance

# mRNAs to RNA-seq fragments




$K_{ij}$ = count of fragments aligned to gene i, sample j

is proportional to:

- expression of RNA
- length of gene
- sequencing depth
- lib. prep. factors (PCR)
- in silico factors (alignment)
- ...

# Expression Index

- RPKM (Reads per kilobase of transcript per million reads of library)
  - Total reads / 1M, divide by gene length in KB
  - Corrects for coverage, gene length
  - TopHat / Cufflinks
  - FPKM (Fragments), PE libraries, ~RPKM/2

- TPM (transcripts per million) RSEM (Li et al, Bioinfo 2011)
  - Divide read count by gene length in KB (RPK) FIRST, divide by scaling factor (sum of RKP across all genes / 1M)
  - Proportion of reads mapped to a gene in each sample is comparable

- CPM (count per million) do not normalize gene length

# RPKM vs TPM

RPKM

| Gene Name | Rep1 RPKM | Rep2 RPKM | Rep3 RPKM |
|---|---|---|---|
| A (2kb) | 1.43 | 1.33 | 1.42 |
| B (4kb) | 1.43 | 1.39 | 1.42 |
| C (1kb) | 1.43 | 1.78 | 1.42 |
| D (10kb) | 0 | 0 | 0.009 |
| Total: | 4.29 | 4.5 | 4.25 |

... the sums of each column are very different.

TPM

| Gene Name | Rep1 TPM | Rep2 TPM | Rep3 TPM |
|---|---|---|---|
| A (2kb) | 3.33 | 2.96 | 3.326 |
| B (4kb) | 3.33 | 3.09 | 3.326 |
| C (1kb) | 3.33 | 3.95 | 3.326 |
| D (10kb) | 0 | 0 | 0.02 |
| Total: | 10 | 10 | 10 |

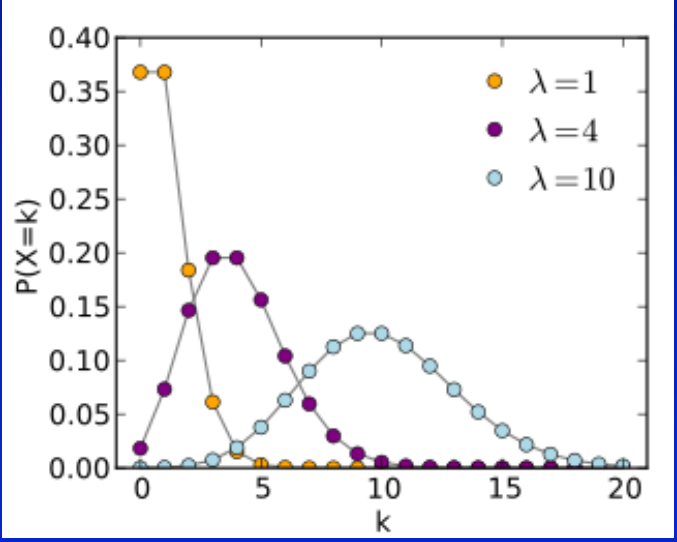
# Differential RNA-seq with DESeq

# Sequencing Read Distribution

- The number of patients arriving in an emergency room between 10 and 11 pm
- # Reads mapped to a gene of 3KB in length
- Poisson dist

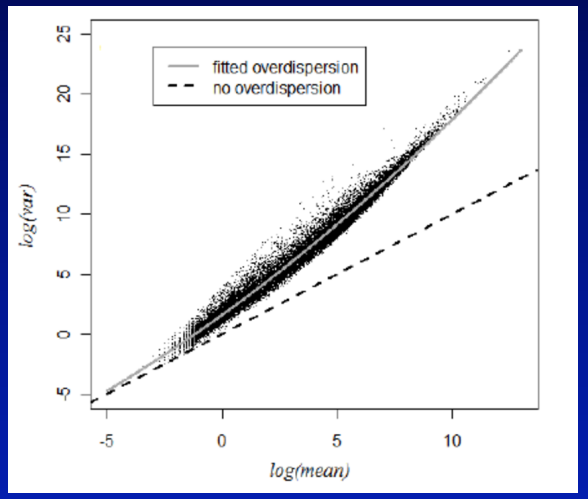
$$P(k \text{ events in interval}) = \frac{\lambda^k e^{-\lambda}}{k!}$$

  - λ average events per interval
  - K # events in an interval
  - Var = mean = λ

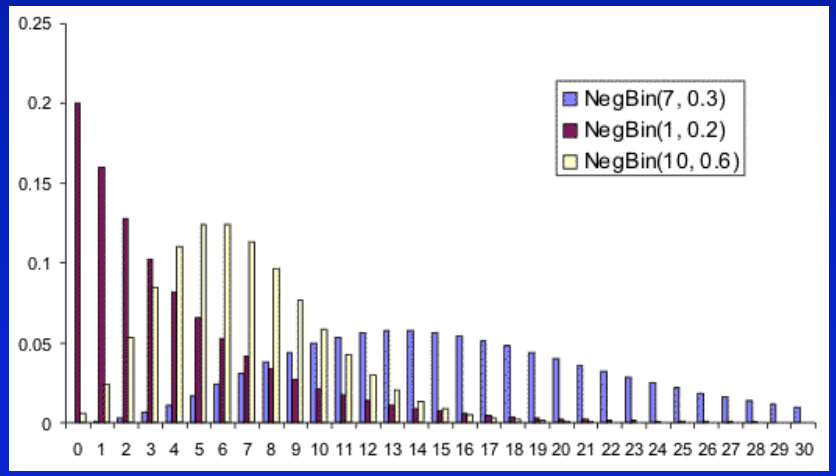


# Sequencing Read Distribution

- In reality, sequencing data is over-dispersed
  - (Mean < Variance)
- Negative binomial
  - NB(r, p)
  - # of success before the first r failure, if Pb(succ) is p




| | |
|---|---|
| **Mean** | $\frac{pr}{1-p}$ |
| **Variance** | $\frac{pr}{(1-p)^2}$ |



# DESeq2: Modeling RNA-seq Read Over Dispersion

raw count for gene i, sample j

normalization factor

quantity of interest

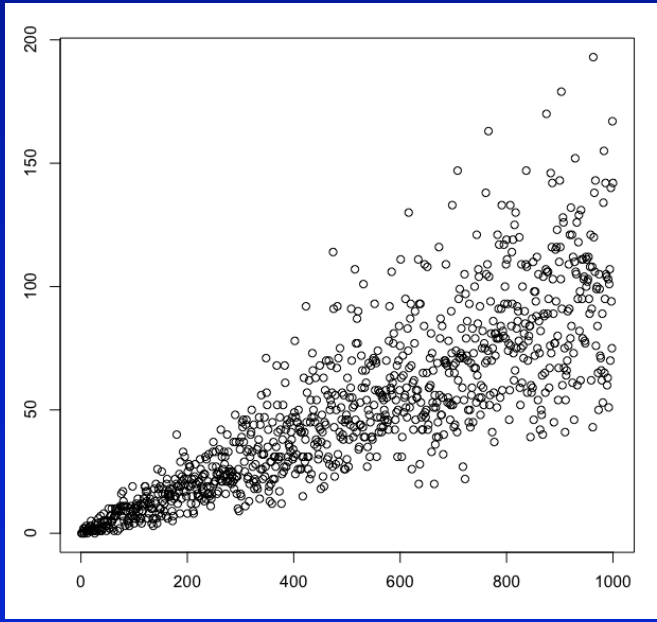
one dispersion per gene

$$K_{ij} \sim \mathrm{NB}(s_{ij} q_{ij}, \alpha_i)$$

$$\mathrm{Var}(K_{ij}) = \mu_{ij} + \alpha_i \mu_{ij}^2$$

Poisson from sampling fragments

Extra variation due to biological variance




Variance estimated by borrowing information from all the genes – hierarchical models

# DESeq2 Differential Expression

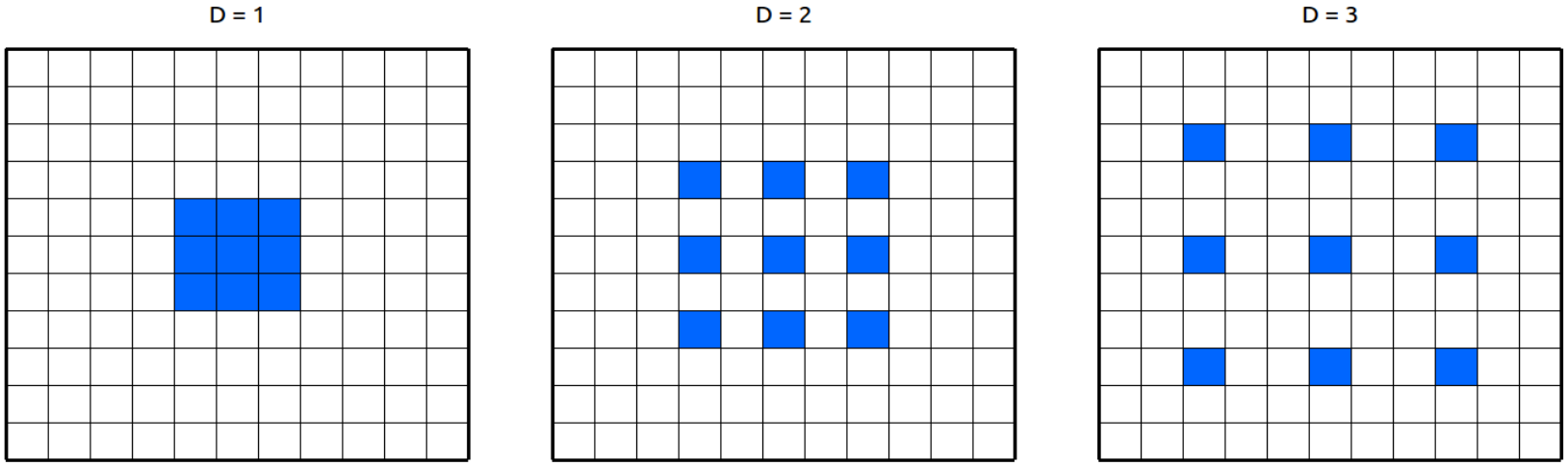
- Normalize raw counts in different libraries

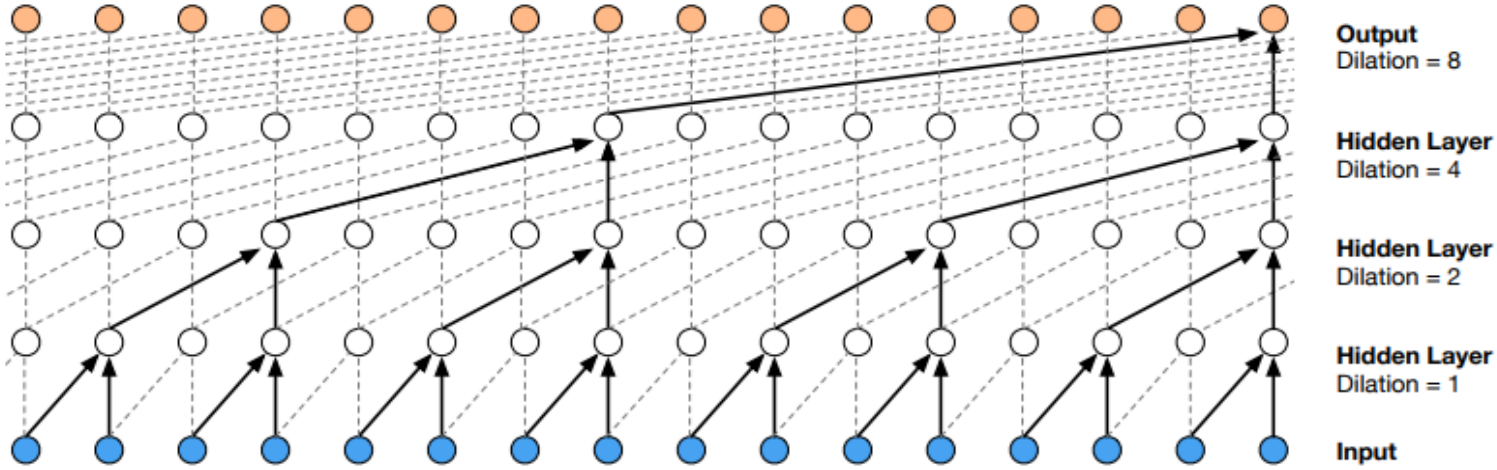
| Gene | Sample #1<br>635 reads | Sample #2<br>635 reads |
|---|---|---|
| A1BG | 30 | 235 |
| A1BG-AS1 | 24 | 188 |
| A1CF | 0 | 0 |
| A2M | 563 | 0 |
| A2M-AS1 | 5 | 39 |
| A2ML1 | 13 | 102 |

- Stabilize / shrink variance by borrowing information from other genes
- Differential expression: test whether gene i expression follows same NB()

$$\Pr(X = k) = \binom{k + \frac{\mu^2}{\sigma^2 - \mu} - 1}{k} \left(\frac{\sigma^2 - \mu}{\sigma^2}\right)^k \left(\frac{\mu}{\sigma^2}\right)^{\mu^2/(\sigma^2 - \mu)}$$

# Dilated Convolution

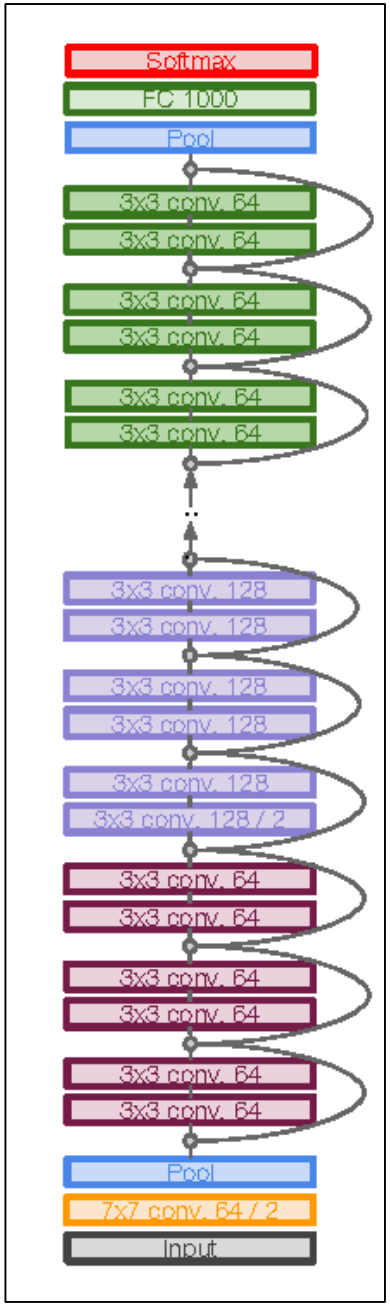


# ResNet

- *Deep Residual Learning for Image Recognition - Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; 2015*
- Extremely deep network – 152 layers
- Deeper neural networks are more difficult to train.
- Deep networks suffer from vanishing and exploding gradients.
- Present a residual learning framework to ease the training of networks that are substantially deeper than those used previously.

**[He et al., 2015]**




# ResNet

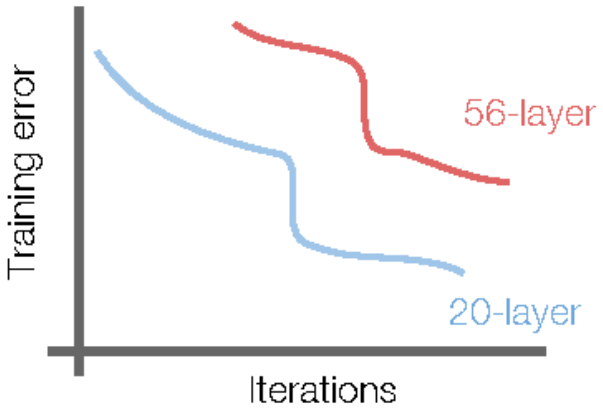
- ILSVRC'15 classification winner (3.57% top 5 error, humans generally hover around a 5-10% error rate)
  Swept all classification and detection competitions in ILSVRC'15 and COCO'15!

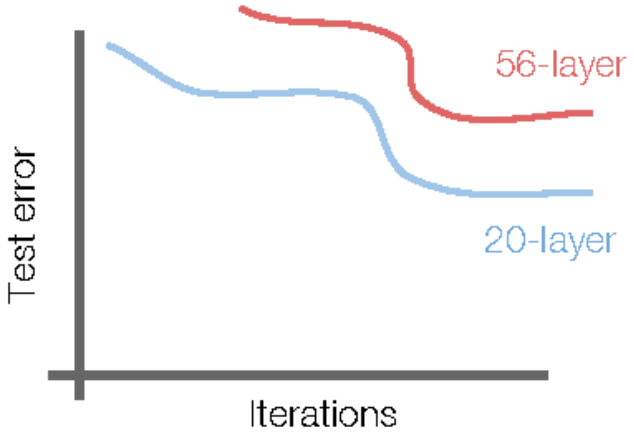
Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

**[He et al., 2015]**

# ResNet

- What happens when we continue stacking deeper layers on a convolutional neural network?




- 56-layer model performs worse on both training and test error

-> The deeper model performs worse (not caused by overfitting)!

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

**[He et al., 2015]**

# ResNet

- **Hypothesis**: The problem is an optimization problem. Very deep networks are harder to optimize.
- **Solution**: Use network layers to fit residual mapping instead of directly trying to fit a desired underlying mapping.

- We will use **skip connections** allowing us to take the activation from one layer and feed it into another layer, much deeper into the network.
- Use layers to fit residual $F(x) = H(x) - x$
  instead of $H(x)$ directly

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

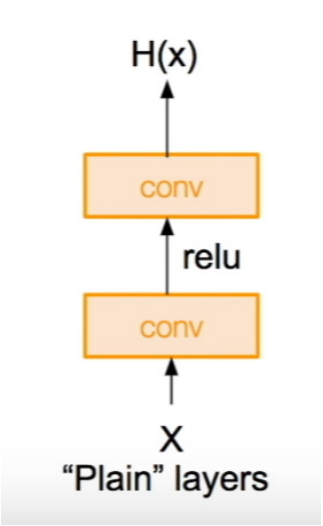
**[He et al., 2015]**

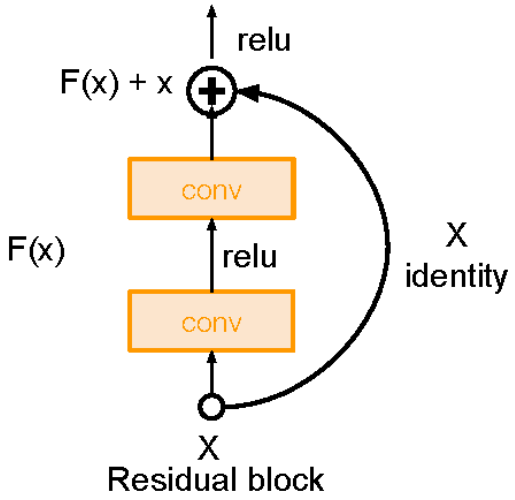
# ResNet

**Residual Block**

Input x goes through conv-relu-conv series and gives us F(x). That result is then added to the original input x. Let's call that H(x) = F(x) + x.

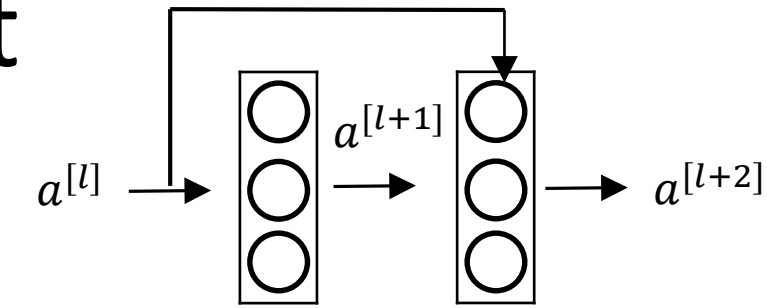
In traditional CNNs, H(x) would just be equal to F(x). So, instead of just computing that transformation (straight from x to F(x)), we're computing the term that we have to *add*, F(x), to the input, x.




**[He et al., 2015]**

# ResNet




Short cut/ skip connection

$a^{[l]}$ → **Linear** → **ReLU** → **Linear** → **ReLU** → $a^{[l+2]}$

$a^{[l+1]}$

$$\mathbf{z}^{[l+1]} = \mathbf{W}^{[l+1]}\,\mathbf{a}^{[l]} + \mathbf{b}^{[l+1]}$$

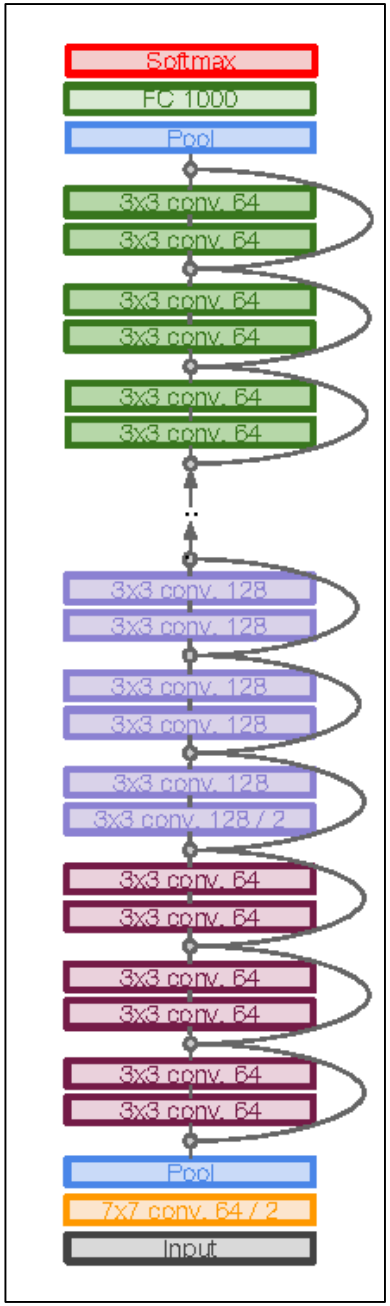
$$\mathbf{a}^{[l+1]} = \mathbf{g}(\mathbf{z}^{[l+1]})$$

$$\mathbf{z}^{[l+2]} = \mathbf{W}^{[l+2]}\mathbf{a}^{[l+1]} + \mathbf{b}^{[l+2]}$$

$$\mathbf{a}^{[l+2]} = \mathbf{g}(\mathbf{z}^{[l+2]})$$

$$\mathbf{a}^{[l+2]} = \mathbf{g}\left(\mathbf{z}^{[l+2]} + \mathbf{a}^{[l]}\right) = \mathbf{g}(\mathbf{W}^{[l+2]}\mathbf{a}^{[l+1]} + \mathbf{b}^{[l+2]} + \mathbf{a}^{[l]})$$

**[He et al., 2015]**




# ResNet

**Full ResNet architecture:**

- Stack residual blocks
- Every residual block has two 3x3 conv layers
- Periodically, double # of filters and downsample spatially using stride 2 (in each dimension)
- Additional conv layer at the beginning
- No FC layers at the end (only FC 1000 to output classes)

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

**[He et al., 2015]**

# ResNet

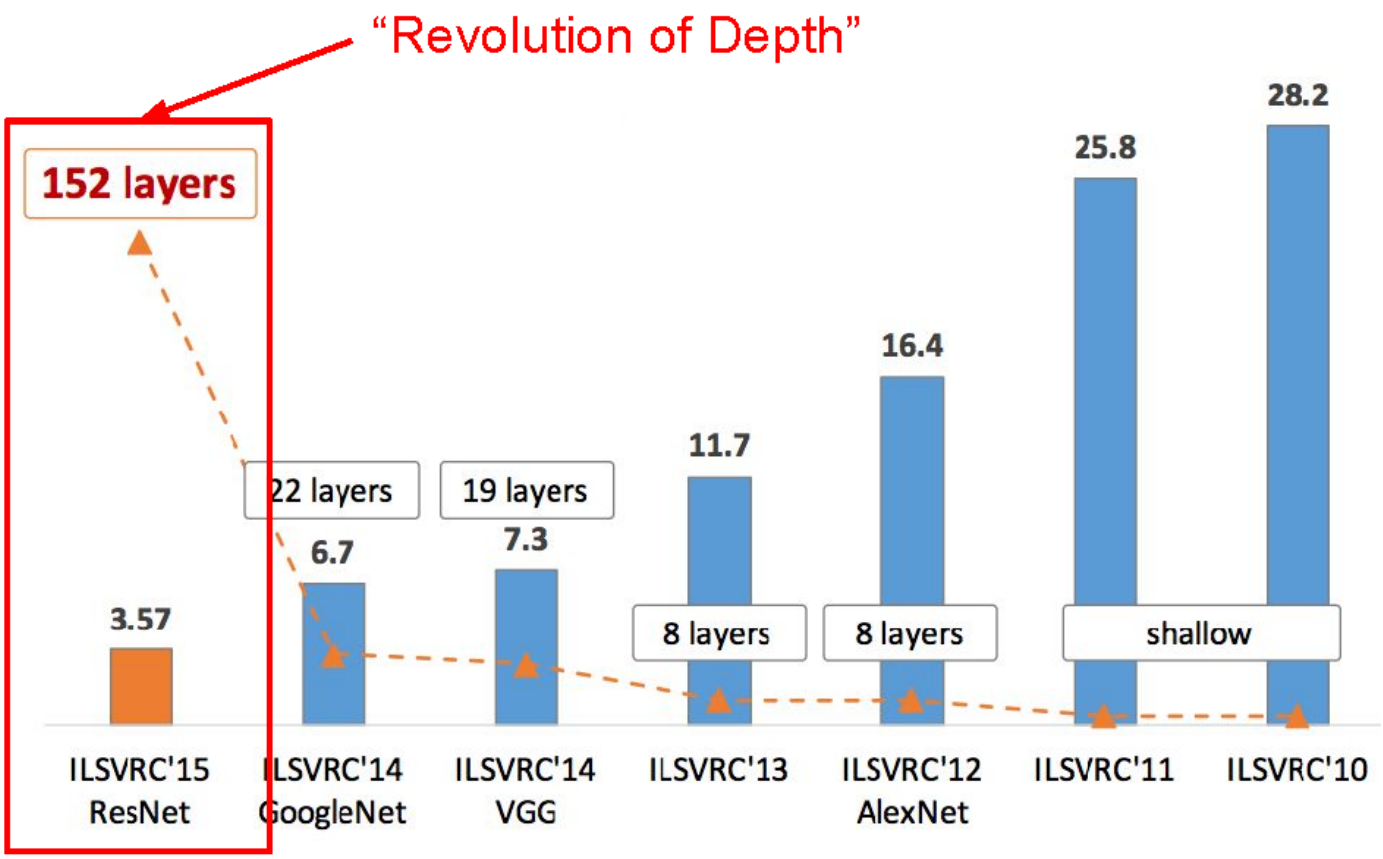
**Experimental Results:**

- Able to train very deep networks without degrading
- Deeper networks now achieve lower training errors as expected

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

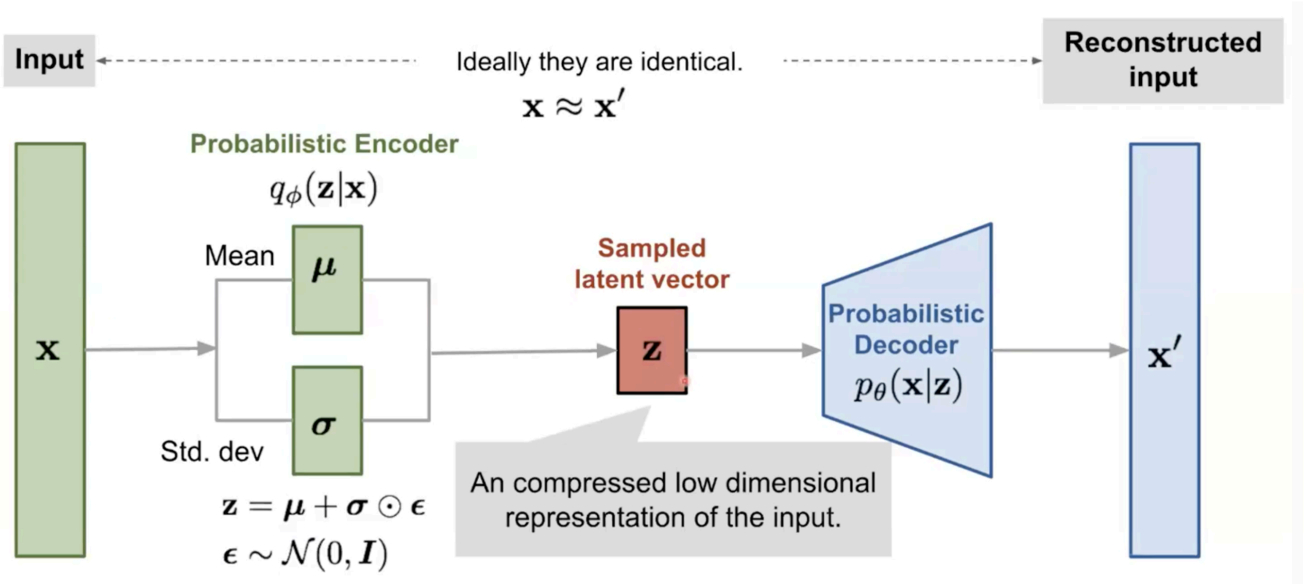
**[He et al., 2015]**

# ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners




Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

# VAE with Gaussian prior, reparameterization trick


https://lilianweng.github.io/lil-log/

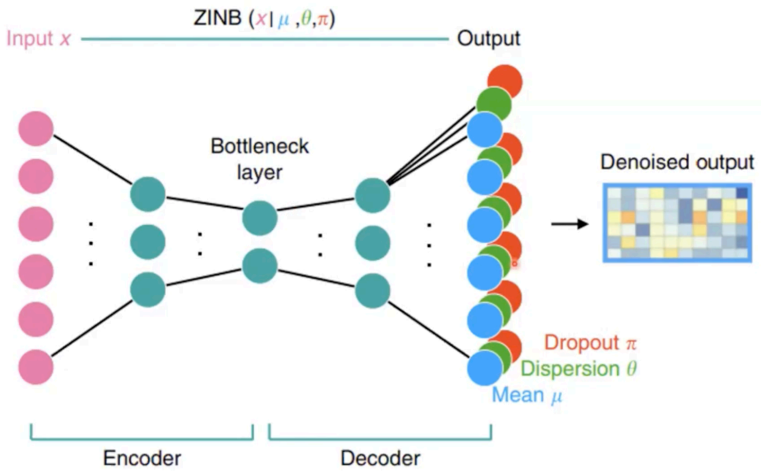
2021-03-19 02:06:45

# DCA—Denoising Count Autoencoder


- Autoencoder (AE) with **Zero-inflated Negative Binomial (ZINB)** loss function
- Negative binomial models the mean $\mu$ and dispersion θ of RNA-seq count
- Zero inflation with a point mass π models the dropout events
- ZINB provides great denoising performance, which benefits downstream analysis, including clustering, time course modeling, differential expression, protein-RNA co-expression and pseudo time analyses.

$$\mathrm{NB}(x;\mu,\theta) = \frac{\Gamma(x+\theta)}{\Gamma(\theta)}\left(\frac{\theta}{\theta+\mu}\right)^{\theta}\left(\frac{\mu}{\theta+\mu}\right)^{x}$$

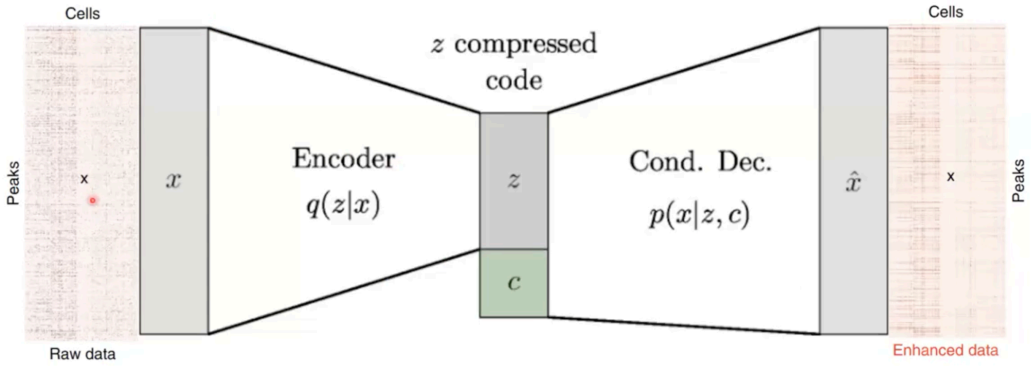
$$\mathrm{ZINB}(x;\pi,\mu,\theta) = \pi\delta_0(x) + (1-\pi)\mathrm{NB}(x;\mu,\theta)$$




Eraslan, Gökcen, et al. "Single-cell RNA-seq denoising using a deep count autoencoder." *Nature communications* 10.1 (2019): 1-14.

2021-03-19 02:08:22

# Conditional VAE


Goal:

To learn a representation informative on biological variations, while remain invariant to confounding factors

Method:

Invariant Coding through VAE

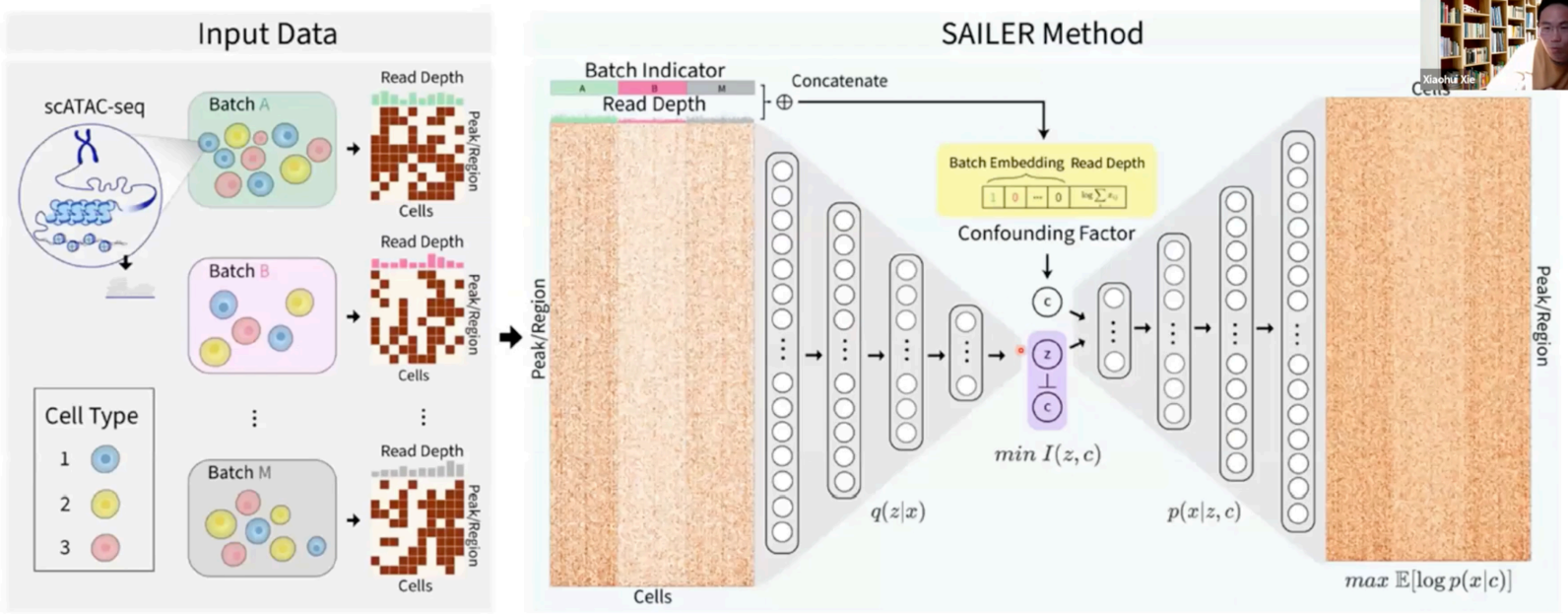
Objective:

Maximize a log-likelihood conditioned on the confounding factors, while minimize the mutual information between latent variable z and confounding factor c.

$$\max \mathbb{E}_{(x,c)}[\log p(x|c)] - \lambda I(z,c).$$

Moyer,D. *et al. NIPS*, 2018

2021-03-19 02:09:50




The overall design of the SAILER method. SAILER takes scATAC-seq data from multiple batches as input. Raw data is pushed through the encoder network to obtain a latent representation. Confounding factors for each single cell are concatenated and fed to the decoder along with the latent representation. Batch information is indicated by a one-hot embedding, and read depth is subject to log transform and standard normalization. To learn a latent representation invariant to changes in confounding factors, mutual information between the latent variables and confounding factors are minimized during training.

# Learning invariant representations


Variational loss

$$L_{\text{VAE}} = \mathbb{E}_{\mathbf{x},\mathbf{c}\sim q(\mathbf{x},\mathbf{c})}\left[-\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z},\mathbf{c})] + D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z}))\right]$$

Minimizing both variation loss and mutual information between latent and conditional variables

$$L_{\text{VAE}} + \lambda I(\mathbf{z};\mathbf{c}) \qquad q_\phi(\mathbf{z},\mathbf{x},\mathbf{c}) = q(\mathbf{x},\mathbf{c})q_\phi(\mathbf{z}|\mathbf{x})$$

Approximation of the loss function:

$$L(\phi,\theta) = \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\left[D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})) + \lambda D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \parallel q_\phi(\mathbf{z}))\right]$$
$$-(1+\lambda)\mathbb{E}_{\mathbf{x},\mathbf{c}\sim q(\mathbf{x},\mathbf{c})}\left[\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z},\mathbf{c})]\right]$$

$$D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \parallel q_\phi(\mathbf{z})) \approx \sum_{\mathbf{x}}\sum_{\mathbf{x}'} D_{\text{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \parallel q_\phi(\mathbf{z}|\mathbf{x}'))$$

Moyer et al, NeurIPS, 2018