# Regularization

**Danna Gurari**
University of Colorado Boulder
Spring 2024

https://dannagurari.colorado.edu/course/neural-networks-and-deep-learning-spring-2024/

# Review

- Last lecture: NN training implementation
  - Motivation
  - Activation functions
  - Initialization
  - Efficient learning
  - Analyzing Loss Curves
  - Programming tutorial

- Assignments (Canvas)
  - Lab assignment 1 was due earlier today
  - Problem set 2 due in one week

- Questions?

# Today’s Topics

- Regularization
- Parameter norm penalty
- Early stopping
- Dataset augmentation
- Dropout
- Batch normalization
- Programming tutorial

# Today's Topics

- Regularization
- Parameter norm penalty
- Early stopping
- Dataset augmentation
- Dropout
- Batch normalization
- Programming tutorial

# What is Regularization?

"any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error."

- Ch. 5.2 of Goodfellow book on Deep Learning

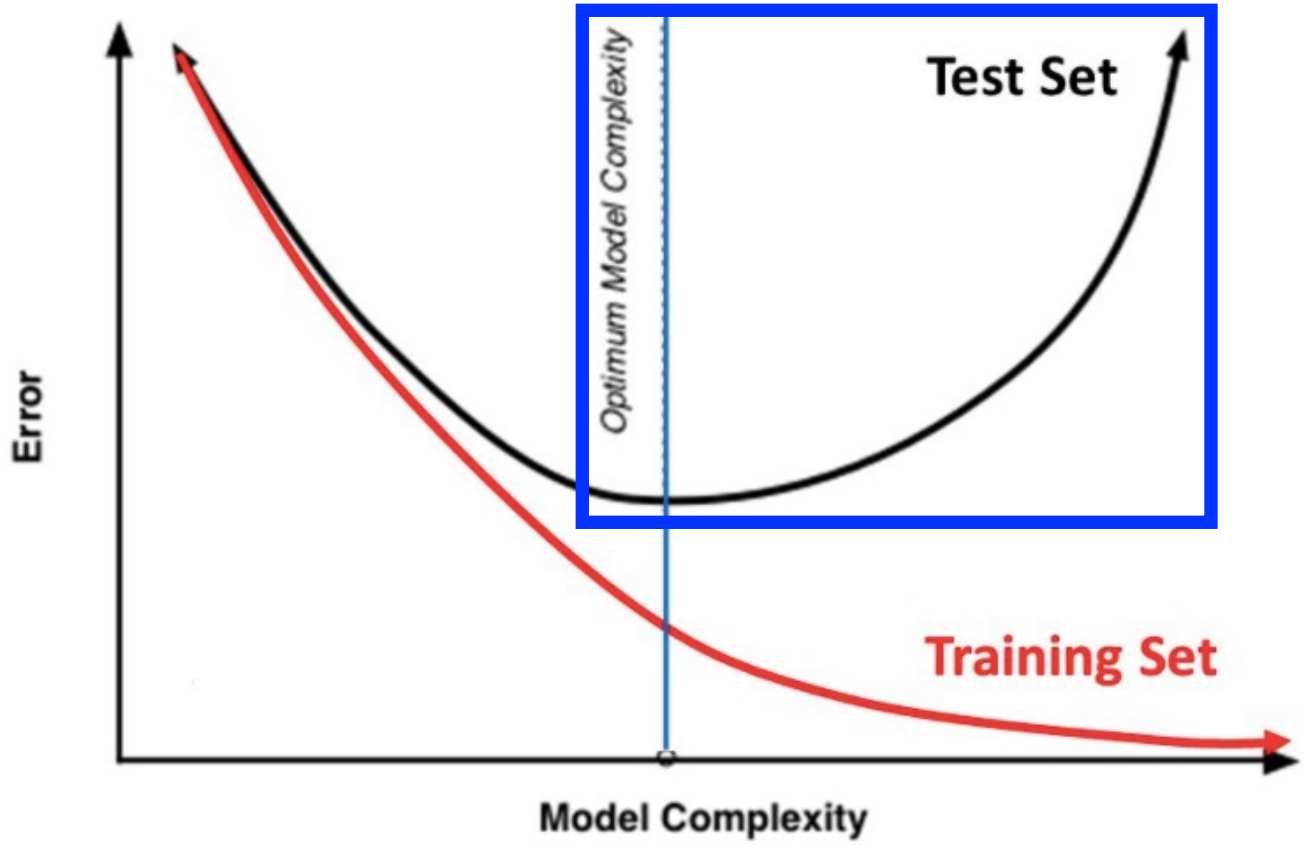
What are strategies for preferring one function over another?

# Today's Topics

- Regularization
- Parameter norm penalty
- Early stopping
- Dataset augmentation
- Dropout
- Batch normalization
- Programming tutorial

# Goal

Rather than exclude functions from a hypothesis space, apply strategies that create a preference for one solution over another to reduce test error




https://cdn.analyticsvidhya.com/wp-content/uploads/2018/04/Screen-Shot-2018-04-04-at-2.43.37-PM.png

# Goal

Rather than exclude functions from a hypothesis space, apply strategies that create a preference for one solution over another to reduce test error; e.g., regularize (c)

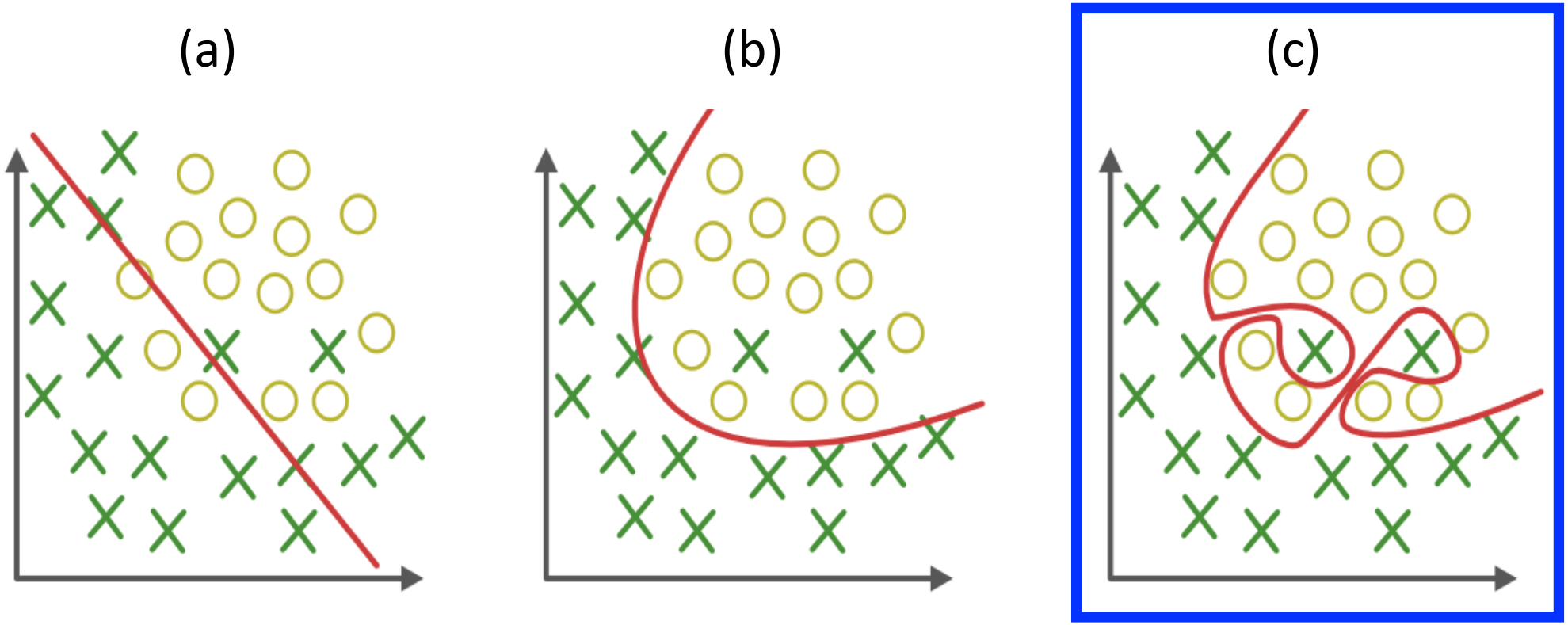
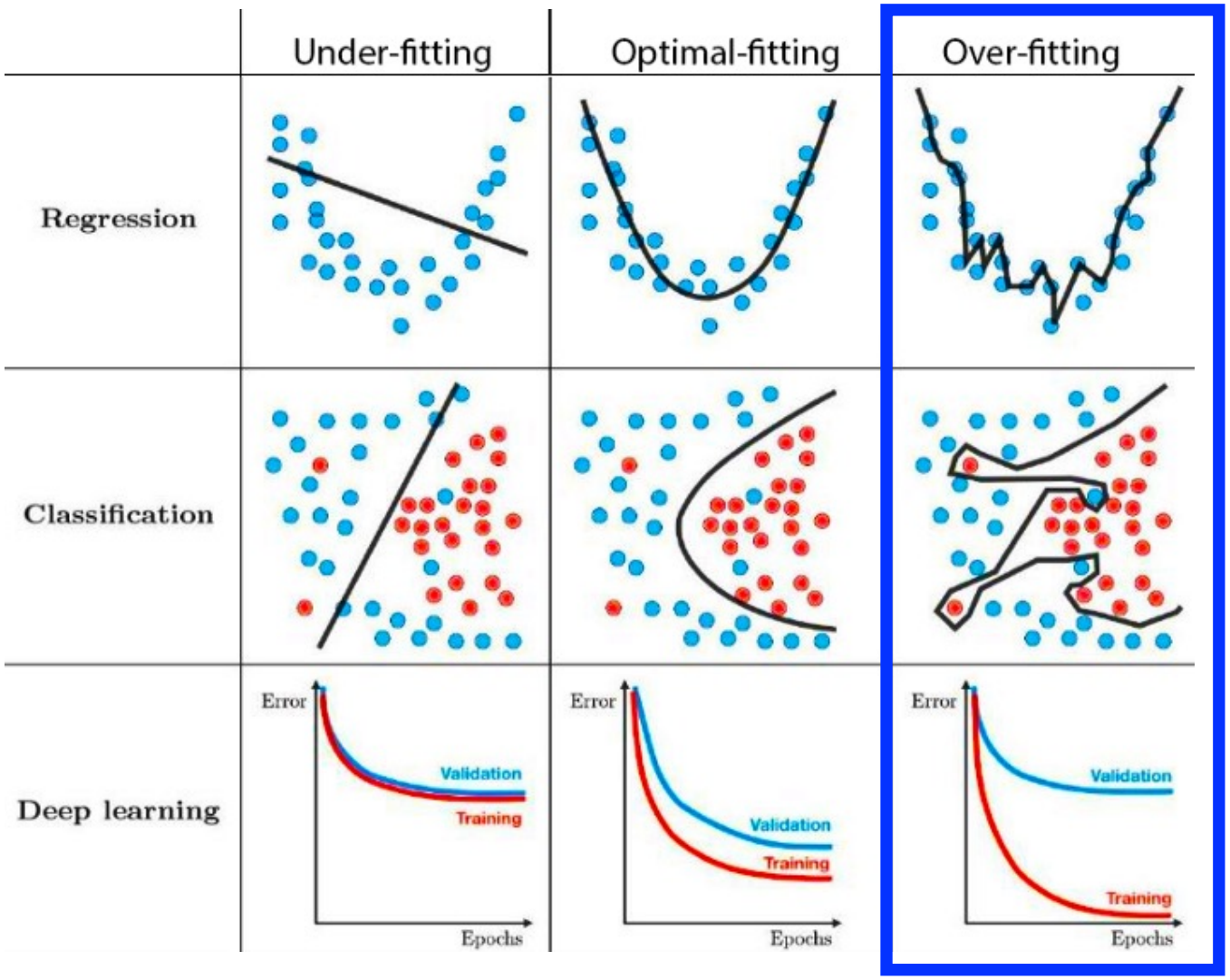



Figure source: https://towardsdatascience.com/underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6fe4a8a49dbf

# Idea: Analogous to Wearing Belt on Big Pants

# Observation: Sign of Overfitting is Large Weights

Very large positive weights get canceled by similarly large negative weights (i.e., due to correlated model parameters) in order to model noise




https://towardsdatascience.com/techniques-for-handling-underfitting-and-overfitting-in-machine-learning-348daa2380b9

# Idea: Penalize Large Weights in Objective Function

e.g., objective is to *minimize* **s**um of **s**quared **e**rrors over training examples

- L2 norm: penalize squared weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 + \alpha \sum_{j=1}^{m} w_j^2$$

- L1 norm: penalize absolute weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 + \alpha \sum_{j=1}^{m} |w_j|$$

- *Note: only weights are penalized, not bias terms (bias terms are fewer/less impactful)*

# Idea: Penalize Large Weights in Objective Function

e.g., objective is to *minimize* **s**um of **s**quared **e**rrors over training examples

- L2 norm: penalize squared weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 + \alpha \sum_{j=1}^{m} w_j^2$$

- L1 norm: penalize absolute weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 + \alpha \sum_{j=1}^{m} |w_j|$$

- Hyperparameter determines contribution of norm penalty term (e.g., belt tightness)

# Idea: Penalize Large Weights in Objective Function

e.g., objective is to *minimize* **s**um of **s**quared **e**rrors over training examples

- L2 norm: penalize squared weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 + \alpha \sum_{j=1}^{m} w_j^2$$

- L1 norm: penalize absolute weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 + \alpha \sum_{j=1}^{m} |w_j|$$

Intuitively, larger alpha values prioritizes having weights closer to 0 instead of minimizing sum of squared errors

- Hyperparameter determines contribution of norm penalty term (e.g., belt tightness)

# Idea: Penalize Large Weights in Objective Function

e.g., objective is to *minimize* **s**um of **s**quared **e**rrors over training examples

- L2 norm: penalize squared weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 + \alpha \sum_{j=1}^{m} w_j^2$$

- L1 norm: penalize absolute weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 + \alpha \sum_{j=1}^{m} |w_j|$$

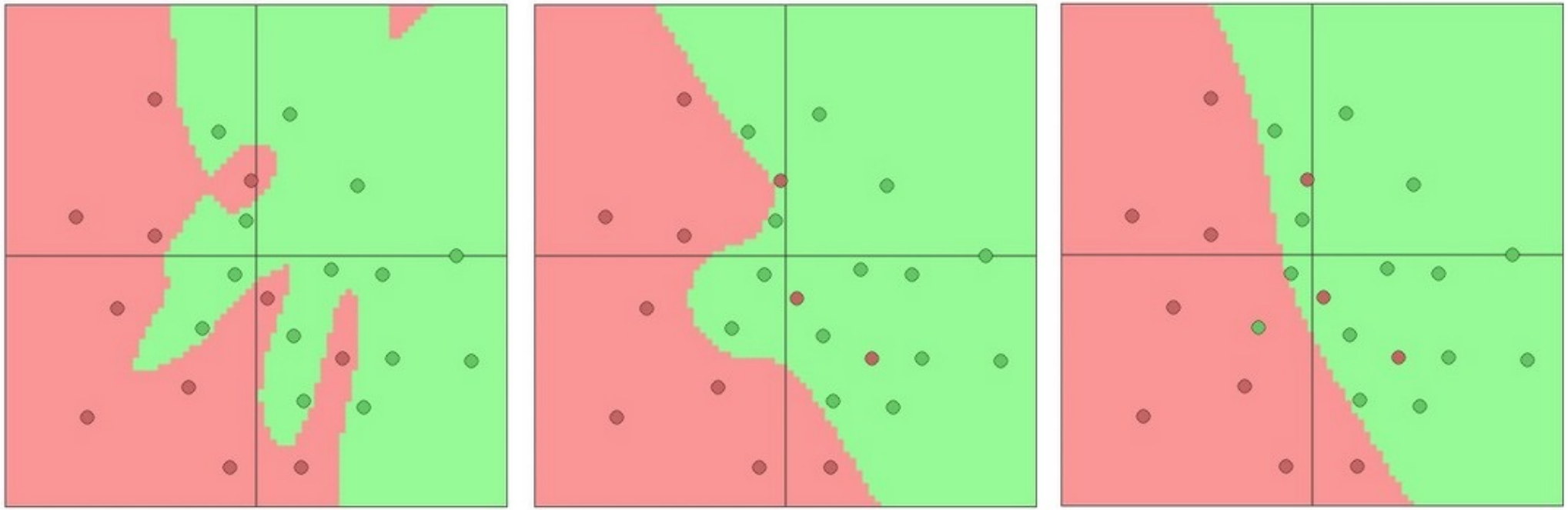
- Gradient derivation for learning with norm penalties found in assigned readings

# How to Set Alpha?

$$Error = \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2 + \alpha \sum_{j=1}^{m} w_j^2$$

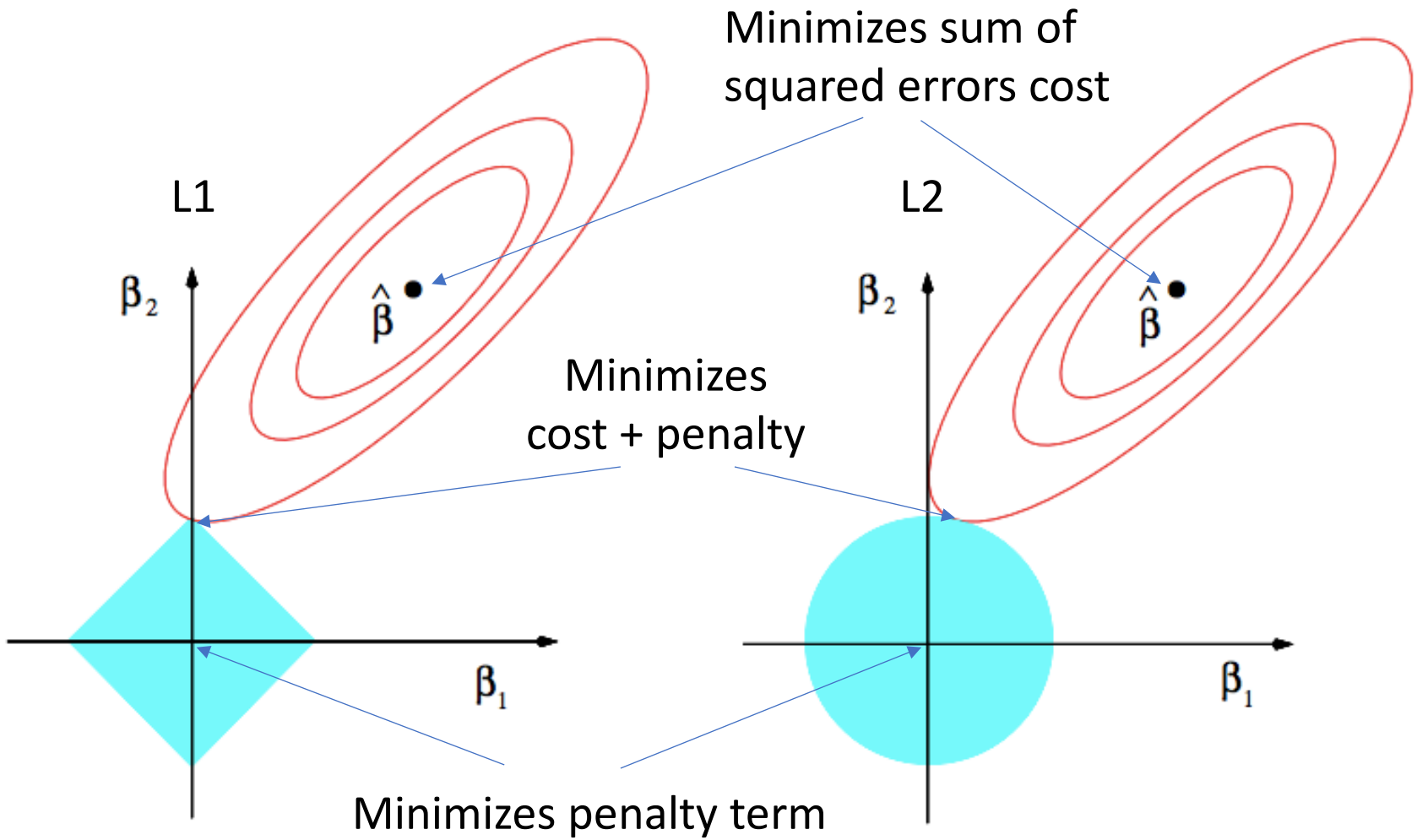
Shown is the same neural network with different levels of regularization. Which model has the largest value for alpha (i.e., largest norm penalty contribution)?

(a) (b) (c)



https://cs231n.github.io/neural-networks-1/

# Geometric Interpretation in 2D




Note: L2 commonly used in practice

Contour of least square error function

Contour of regularization constraint functions shown below both plots

https://web.stanford.edu/~hastie/Papers/ESLII.pdf

# Implementation Detail: Can Penalize Weights Globally as Well As Per Layer

e.g., objective is to *minimize* **s**um of **s**quared **e**rrors over training examples

- L2 norm: penalize squared weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2 + \alpha \boxed{\sum_{j=1}^{m} w_j^2}$$

- L1 norm: penalize absolute weight values

$$Error = \sum_{i=1}^{n} (y^{(i)} - \widehat{y}^{(i)})^2 + \alpha \boxed{\sum_{j=1}^{m} |w_j|}$$

- *Note: only weights are penalized, not bias terms*

# Today’s Topics

- Regularization
- Parameter norm penalty
- Early stopping
- Dataset augmentation
- Dropout
- Batch normalization
- Programming tutorial

# Recall: Overfitting Solution is Early Stopping

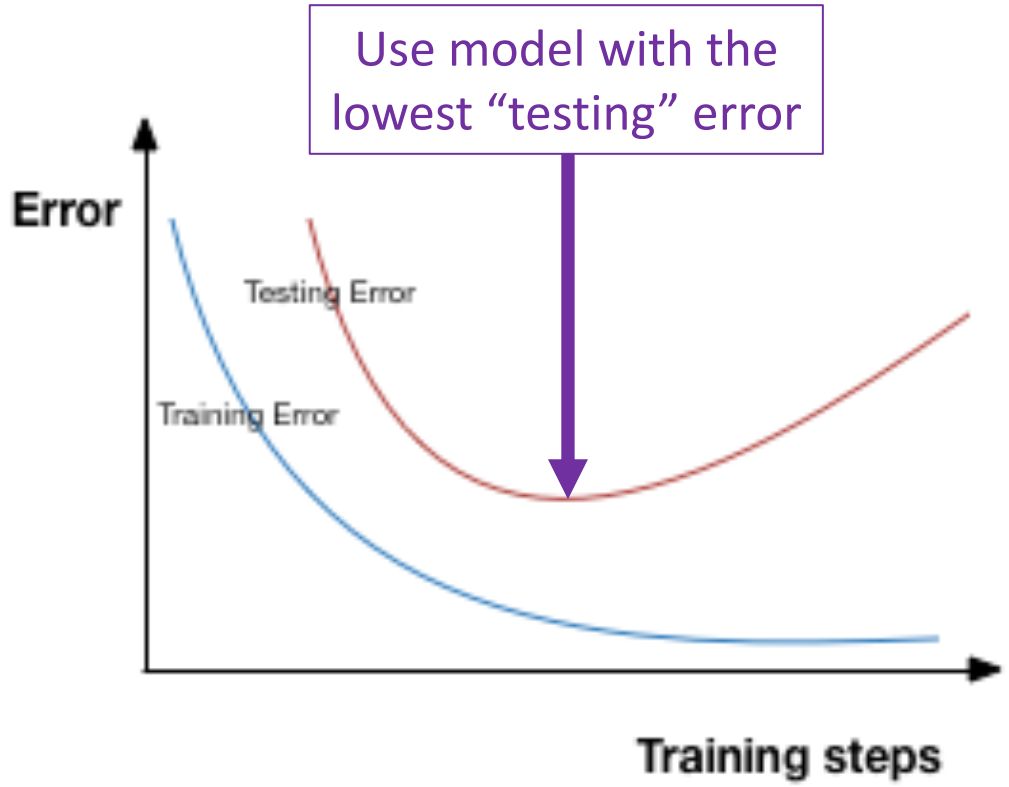
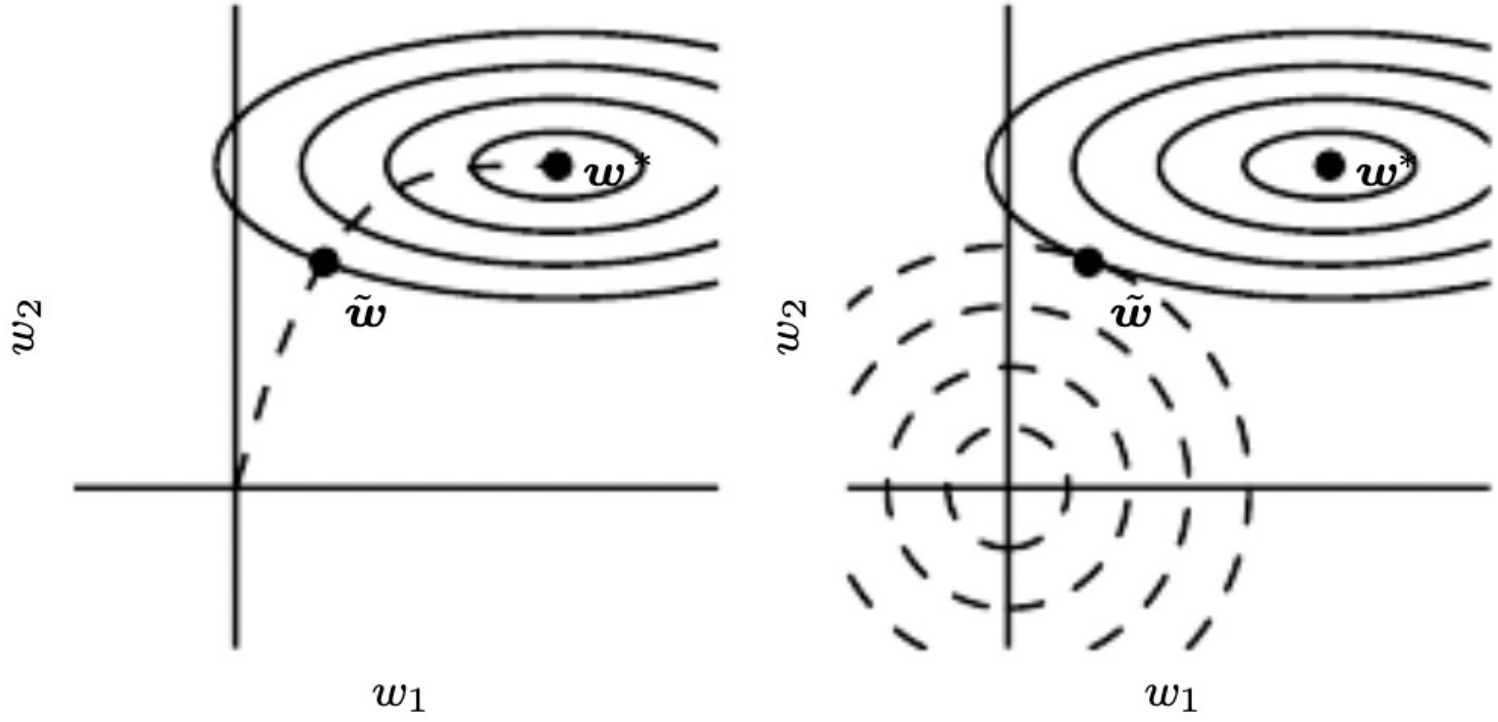



Image Source: https://chatbotslife.com/regularization-in-deep-learning-f649a45d6e0

# Why Early Stopping Acts As a Regularizer

With parameters initialized around the origin, early stopping can behave like a parameter norm penalty (e.g., L2, without having a hyperparameter to tune) since weight values have an insufficient training duration to grow too large; e.g.,



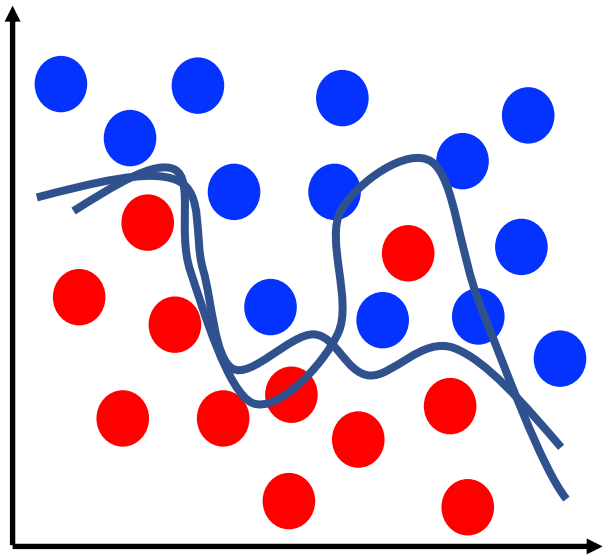

https://www.deeplearningbook.org/contents/regularization.html

# Today's Topics

- Regularization
- Parameter norm penalty
- Early stopping
- Dataset augmentation
- Dropout
- Batch normalization
- Programming tutorial

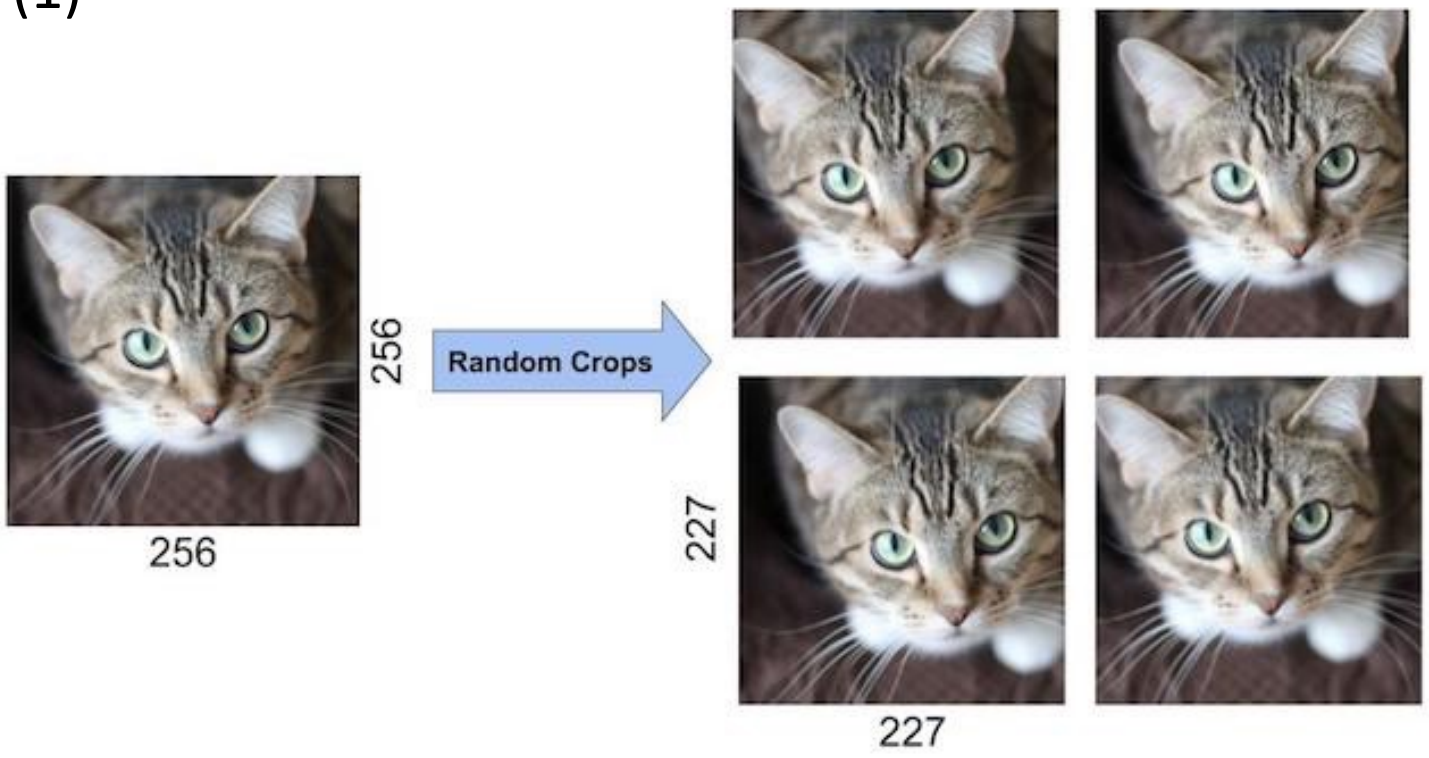
# Recall: Overfitting Solution is to Add Data

Adding training data



# Data Augmentation Strategies; e.g., images

(1)




(2)

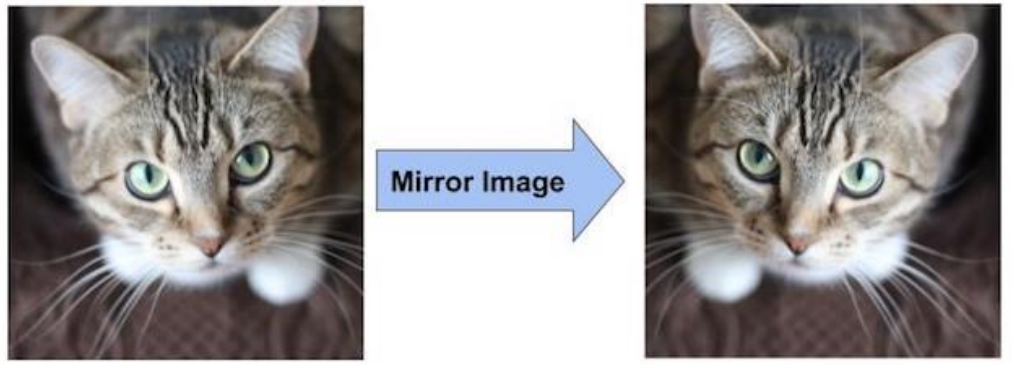
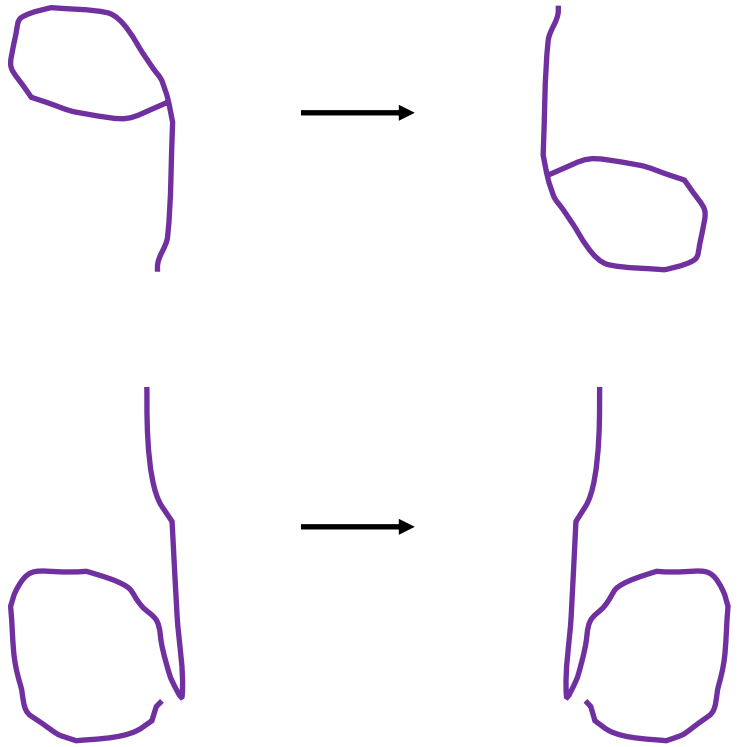



Figure Source: https://learnopencv.com/understanding-alexnet/

# Data Augmentation Strategies; e.g., images

- Caution: augmentation scheme should not conflict with target application; e.g., image mirroring and flipping could be poor choices for character recognition



# Today's Topics

- Regularization
- Parameter norm penalty
- Early stopping
- Dataset augmentation
- Dropout
- Batch normalization
- Programming tutorial

# Idea: Use Wisdom of the Crowds

More than 1: Ensemble

# Why Choose Ensemble vs One Predictor?

- Reduces probability for making a wrong prediction
- Suppose:
  - n classifiers for binary classification task
  - Each classifier has same error rate $\varepsilon$
  - Classifiers are independent (not true in practice!)
  - Probability mass function indicates the probability of error from an ensemble:

Number of classifiers

Classifier error rate

$$P(y \geq k) = \sum_{k}^{n} \binom{n}{k} \varepsilon^k (1-\varepsilon)^{n-k} = \varepsilon_{ensemble}$$

Error probability

# ways to choose *k* subsets from set of size *n*

  - e.g., n = 11, $\varepsilon$ = 0.25; k = 6: probability of error is ~0.034 which is much lower than probability of error from a single algorithm (0.25)

# How to Produce an Ensemble? - Bagging

## Bootstrap Aggregation (1994)

Train algorithm repeatedly on different random subsets of the training set

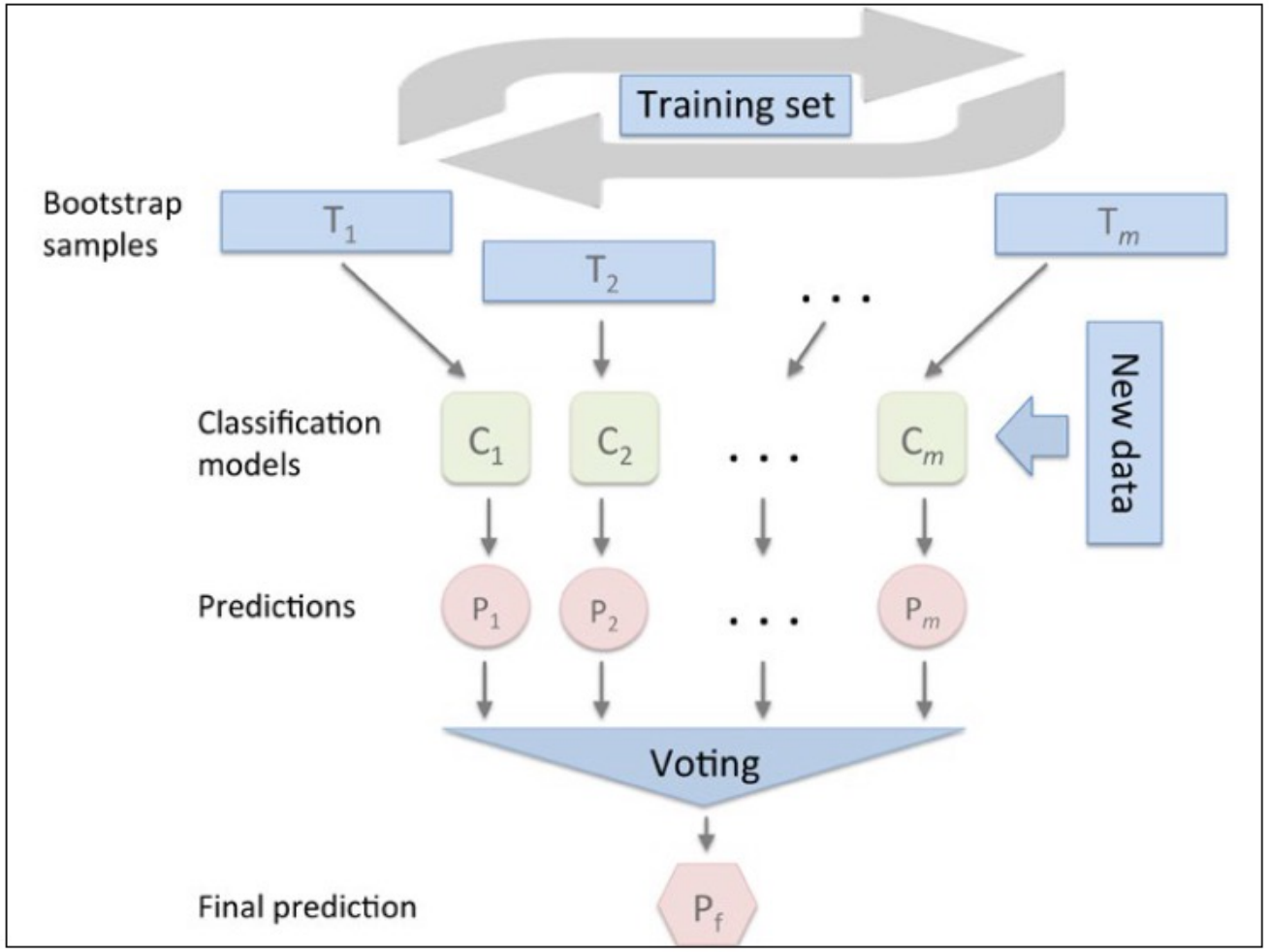



Figure Credit: Raschka & Mirjalili, Python Machine Learning.

# How to Produce an Ensemble? - Bagging

- Build ensemble from "bootstrap samples" drawn with replacement
- e.g.,

Duplicate data can occur for training

Some examples missing from training data; e.g., round 1

| Sample indices | Bagging round 1 | Bagging round 2 | … |
|---|---|---|---|
| 1 | 2 | 7 | … |
| 2 | 2 | 3 | … |
| 3 | 1 | 2 | … |
| 4 | 3 | 1 | … |
| 5 | 7 | 1 | … |
| 6 | 2 | 7 | … |
| 7 | 4 | 7 | … |

Each classifier trained on different subset of data

$C_1$ $C_2$ $C_m$

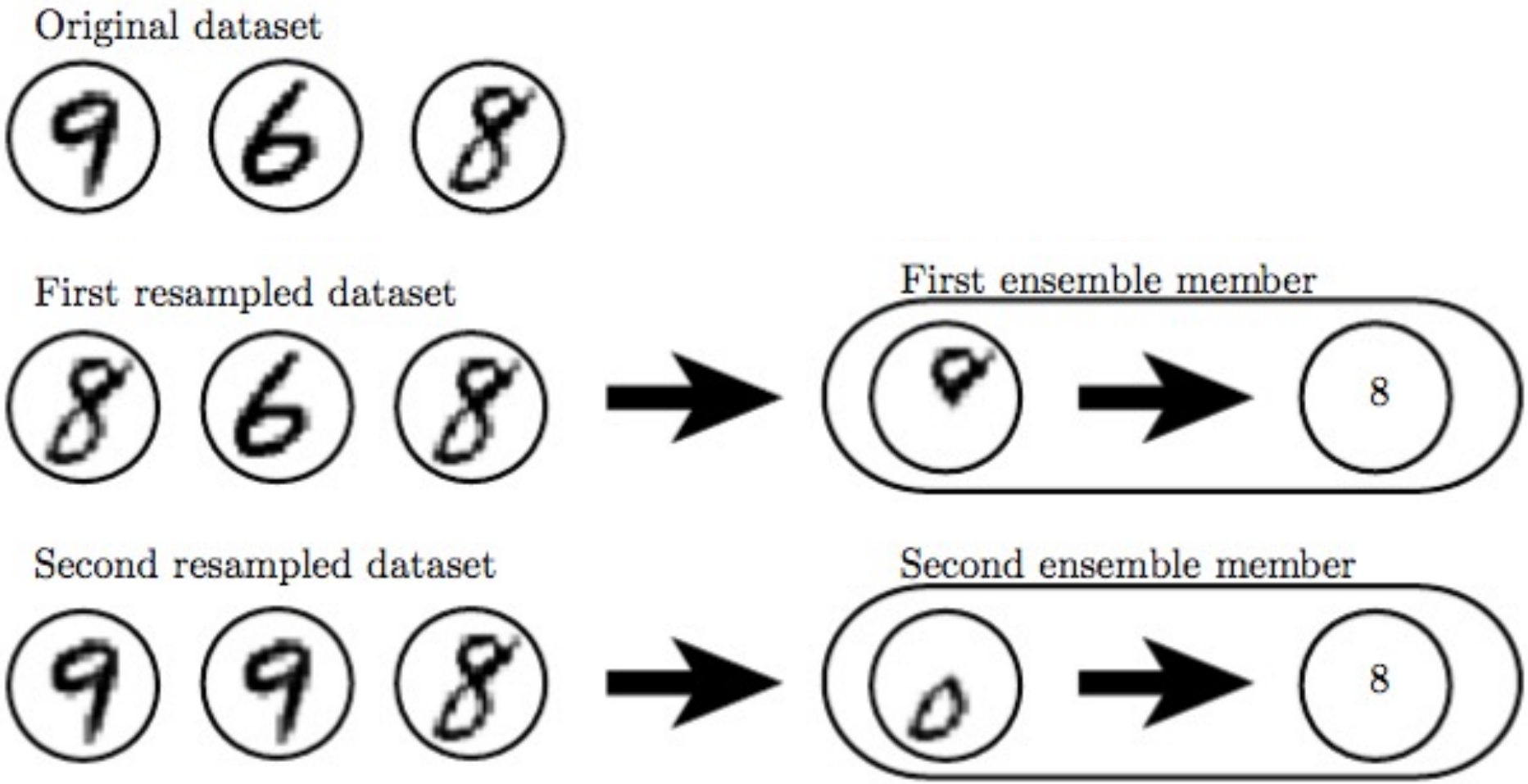
Predictions made from votes by classifiers

Breiman, Bagging Predictors, 1994.

Ho, Random Decision Forests, 1995.

Figure Credit: Raschka & Mirjalili, Python Machine Learning.

# Intuition of Bagging (Train an 8 detector)




Goodfellow et al., Deep Learning (chapter 7), 2016.

# Bagging Limitations

Train algorithm repeatedly on different random subsets of the training set

Why is bagging a poor approach for neural networks?

- Finding optimal hyperparameters for each architecture is time-consuming
- Applying multiple neural networks is often infeasible since the models require lots of memory and are computationally expensive to run

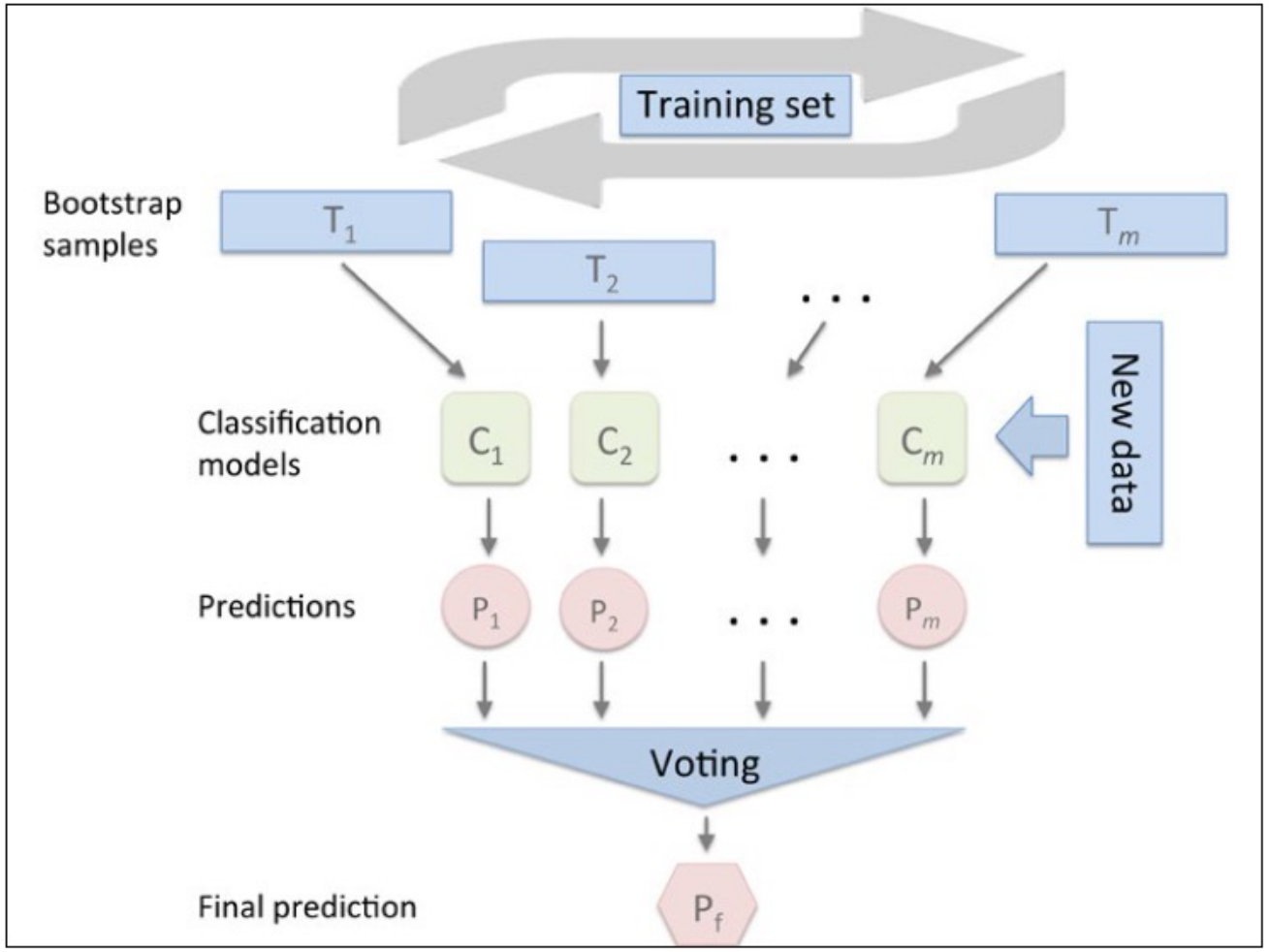
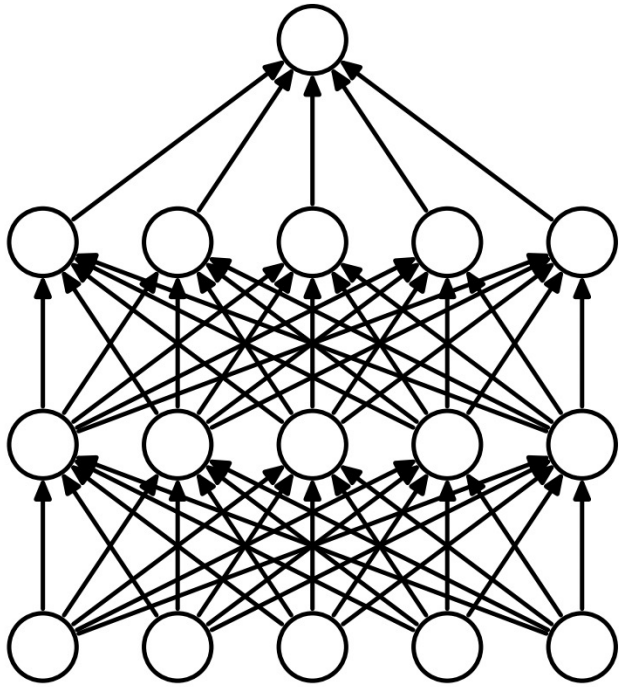


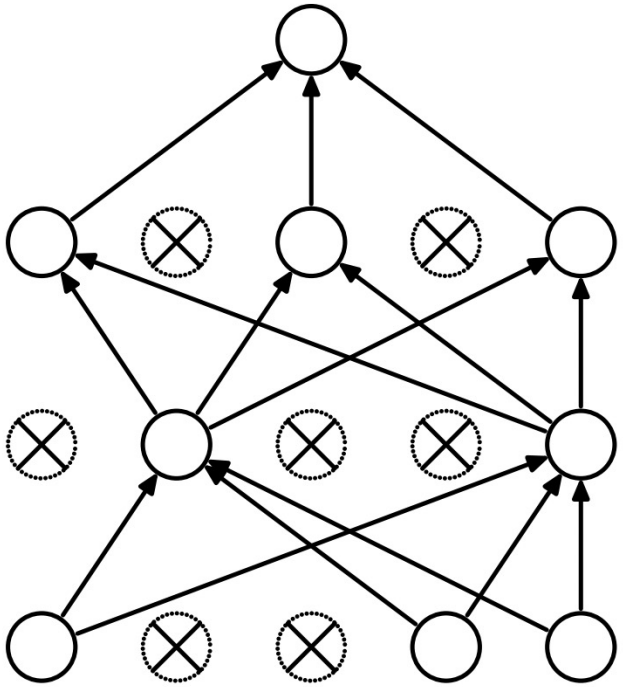

Figure Credit: Raschka & Mirjalili, Python Machine Learning.

# How to Produce an Ensemble?

- Idea: approximate bagging with dropout during training so different sub-models in the network are trained with different training data



(a) Standard Neural Net



(b) After applying dropout.

e.g., drop 50% of units in hidden layers

e.g., drop 20% of units in input layers

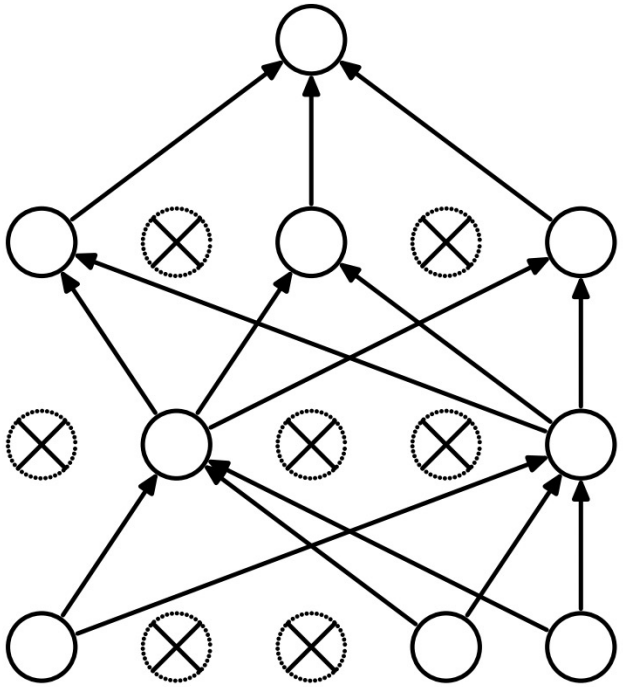
Srivastava et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014

# How to Produce an Ensemble?

- Idea: approximate bagging with dropout during training so different sub-models in the network are trained with different training data

For training, the forward pass and backpropagation run only through the sub-network (with a different dropout per minibatch).

Note dropout can lead to bouncier loss curves since the underlying network continuously changes.



(b) After applying dropout.

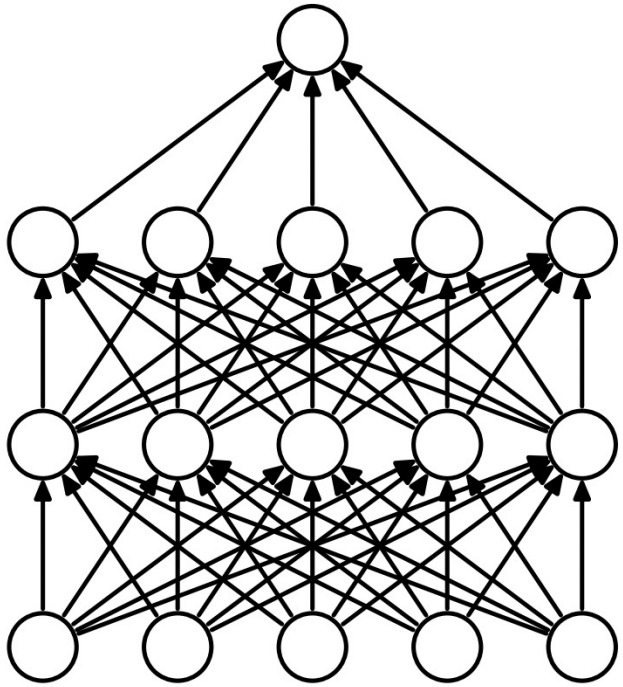
Srivastava et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014

# How to Produce an Ensemble?

- Idea: approximate bagging with dropout during training so different sub-models in the network are trained with different training data

Ensemble is emulated at test time by applying the network without dropout.

To reflect the network's expectation for a smaller activation signal than observed at test time (e.g., input from 2 versus 5 units), each unit's outgoing weights should be multiplied by the probability it was dropped at training.



(b) After applying dropout.

Srivastava et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014

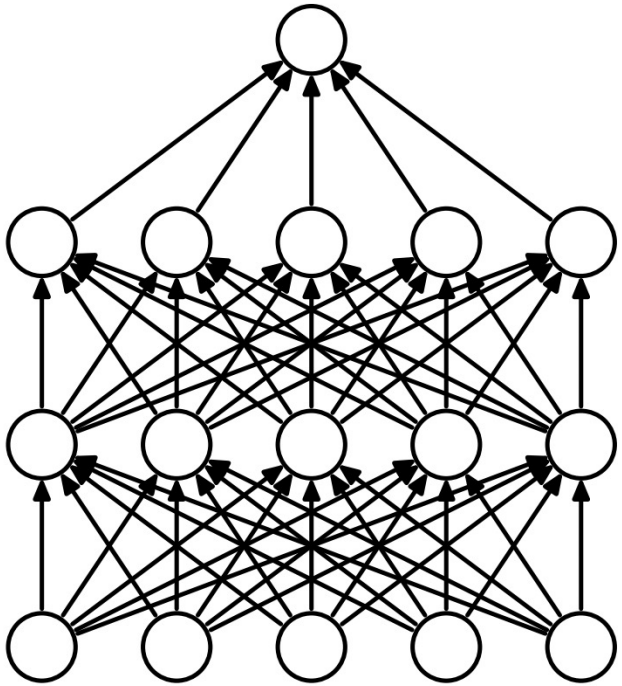
(2012, arXiv)

# Improving neural networks by preventing co-adaptation of feature detectors

G. E. Hinton*, N. Srivastava, A. Krizhevsky, I. Sutskever and R. R. Salakhutdinov
Department of Computer Science, University of Toronto,
6 King's College Rd, Toronto, Ontario M5S 3G4, Canada

# Dropout vs Bagging

- Dropout approximates bagging with many models inexpensively
  - Trains algorithm repeatedly on different random subsets of the training set
- Dropout differences are that subnetworks are not:
  - Trained to convergence (instead, trained for one step)
  - Independent (instead, they all share parameters)

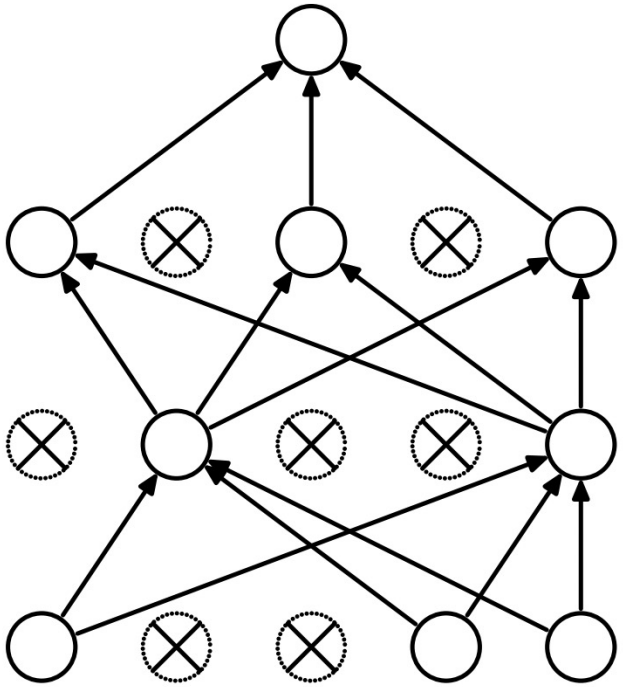


(b) After applying dropout.

Srivastava et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014

# Motivation for Dropout

This approach was motivated by the role of sex in evolution. "… the role of sexual reproduction is not just to allow useful new genes to spread throughout the population, but also to facilitate this process by reducing complex co-adaptations that would reduce the chance of a new gene improving the fitness of an individual."



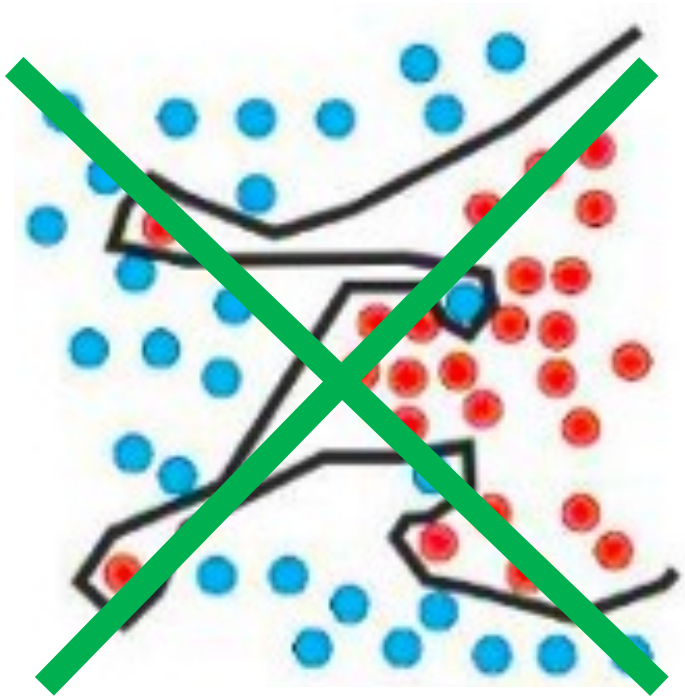
(b) After applying dropout.

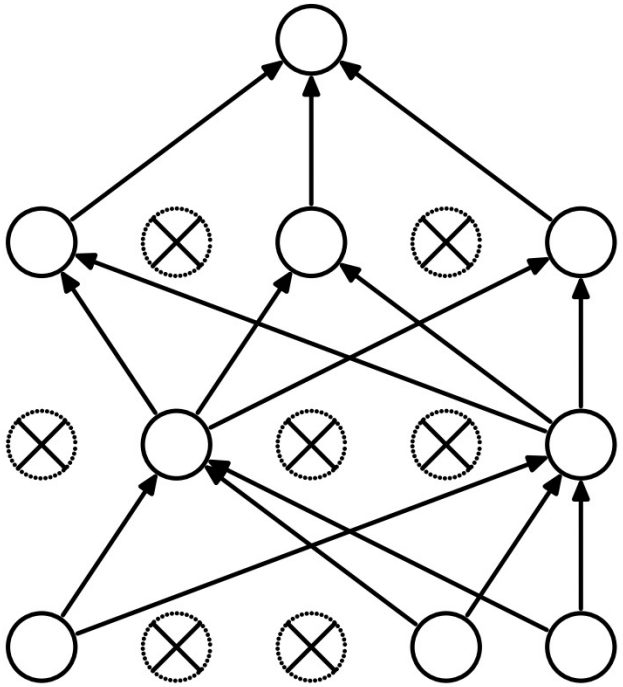
"Similarly, each hidden unit in a neural network trained with dropout must learn to work with a randomly chosen sample of other units. This should make each hidden unit more robust and drive it towards creating useful features on its own without relying on other hidden units to correct its mistakes."

Srivastava et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014

# Motivation for Dropout

Units in the network learn to be useful with many different subsets of other units rather than in conjunction with other units; e.g., mitigates the situation where large positive weights cancel similarly large negative weights, a sign of overfitting.





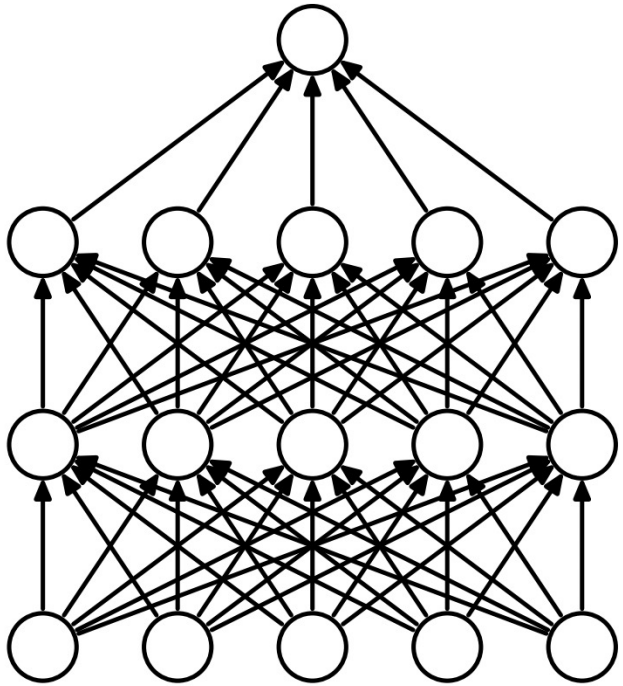
(b) After applying dropout.

https://towardsdatascience.com/techniques-for-handling-underfitting-and-overfitting-in-machine-learning-348daa2380b9

Srivastava et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014

# How to Produce an Ensemble?

A generalization of zeroing units out is to instead multiply units by noise



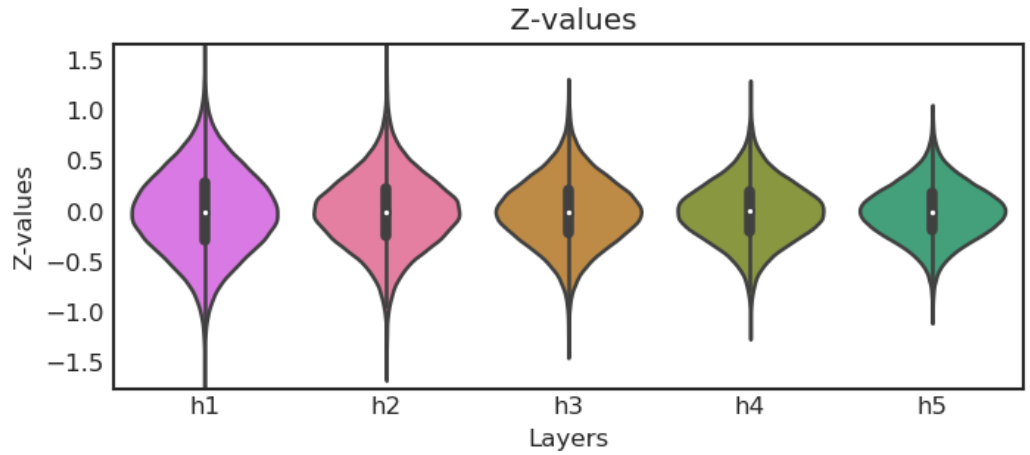
(b) After applying dropout.

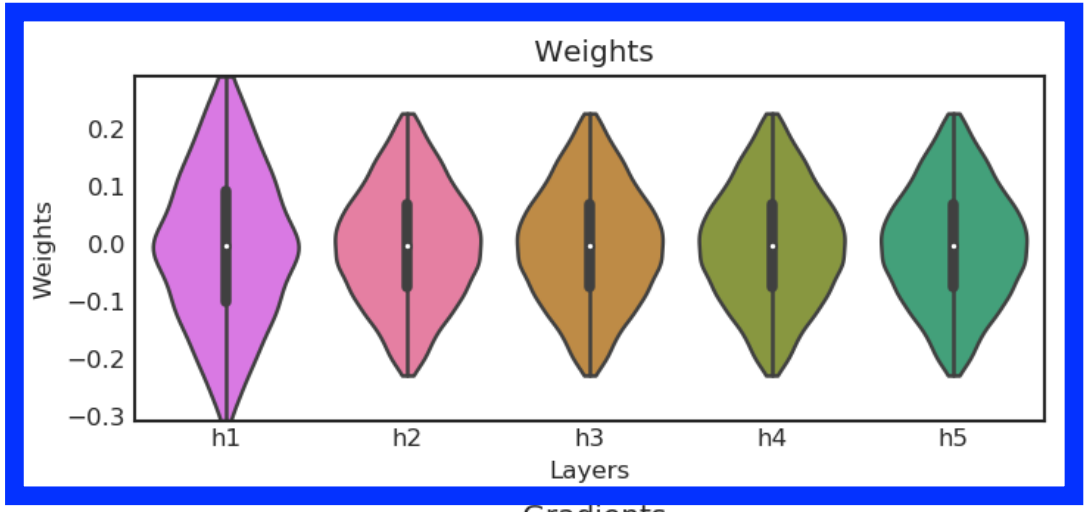
Relevant articles:
*https://towardsdatascience.com/dropout-on-convolutional-layers-is-weird-5c6ab14f19b2
*Wu and Gu. “Towards dropout training for convolutional neural networks.” Neural Networks, 2015.
Srivastava et al. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014

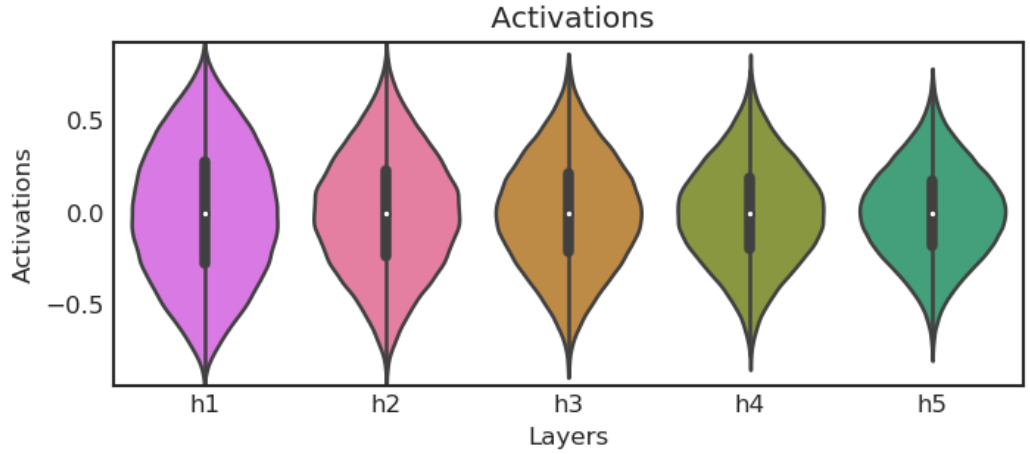
# Today's Topics

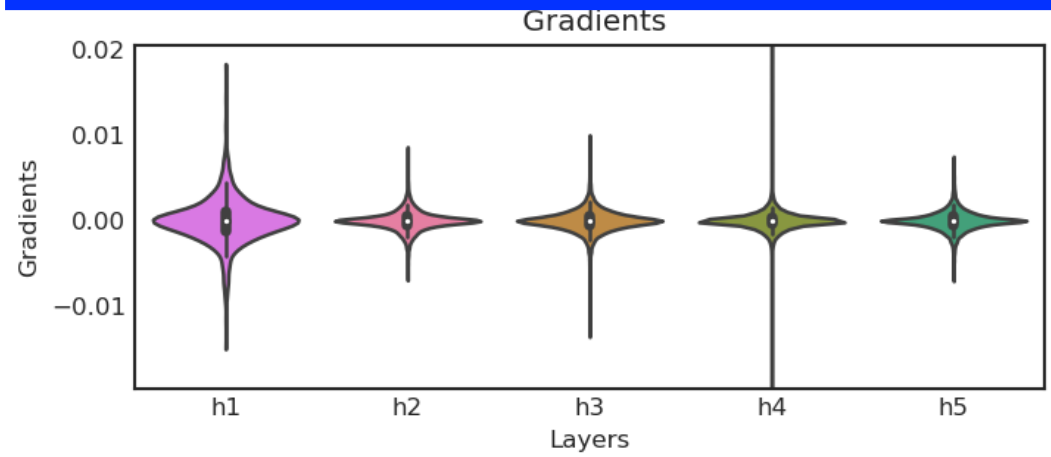
- Regularization
- Parameter norm penalty
- Early stopping
- Dataset augmentation
- Dropout
- Batch normalization
- Programming tutorial

# Recall Model Initialization: Set Weights So Weighted Sum Supports Learning



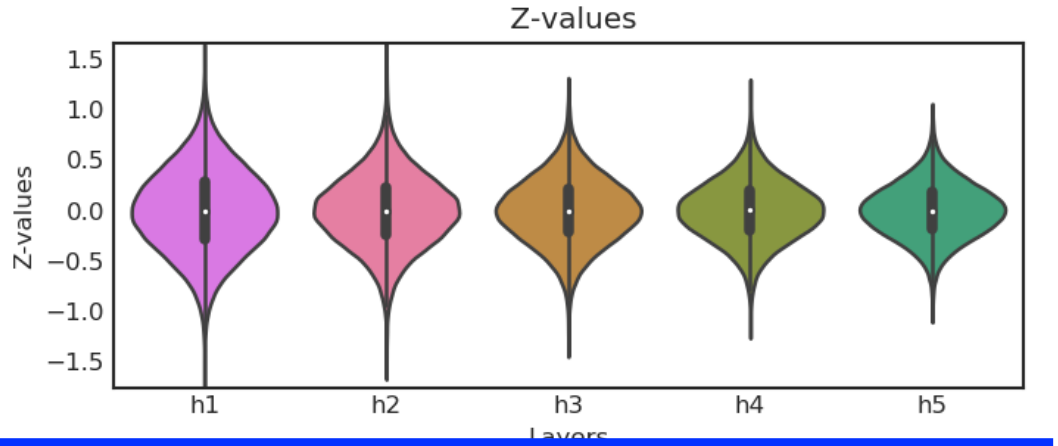



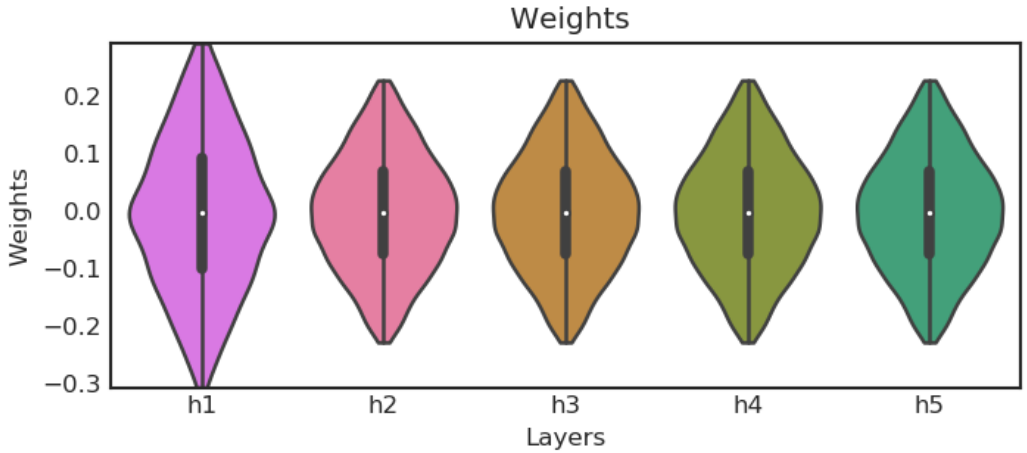



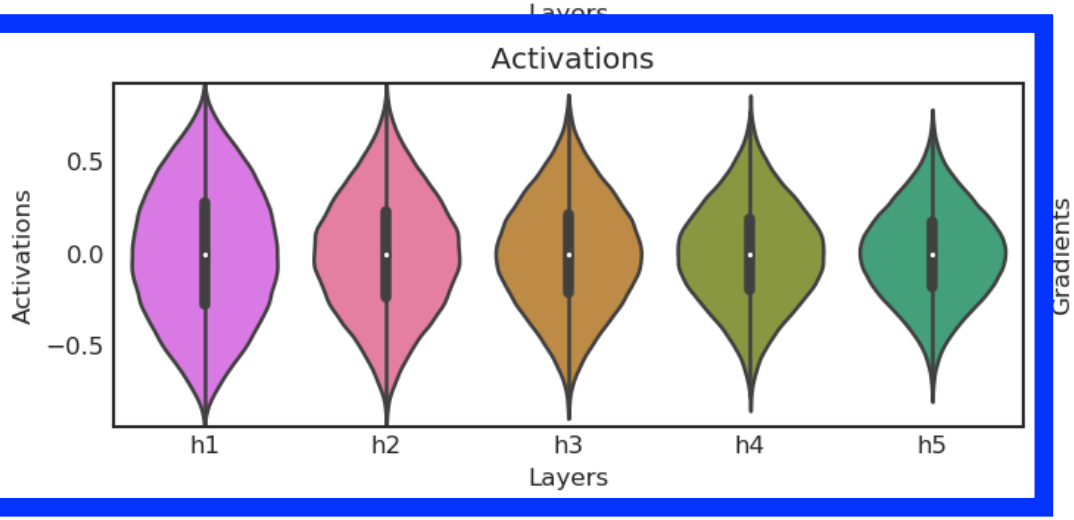

Weights set so **z-values** have similar variance across layers in range of (1, -1), **activations** are similar across layers without going to extreme large/small values, and consequently **gradients** have similar variance across layers that support learning

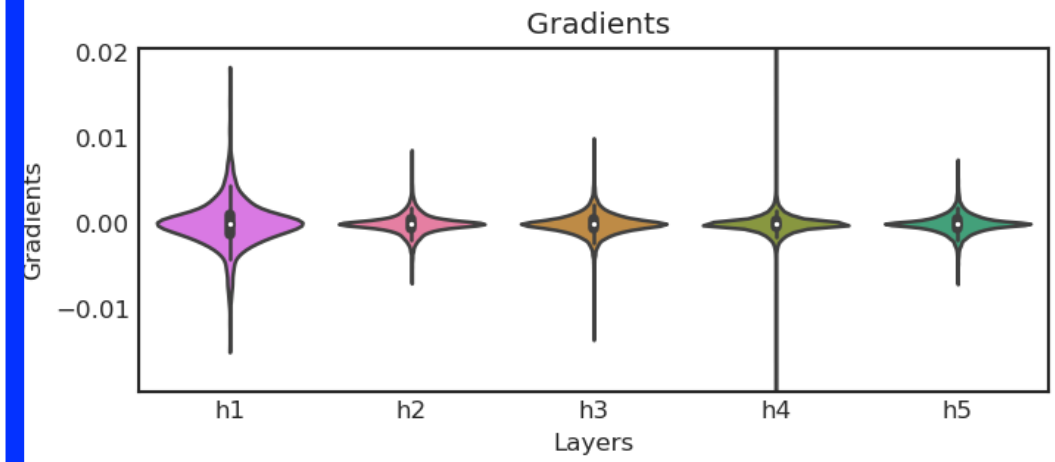
https://towardsdatascience.com/hyper-parameters-in-action-part-ii-weight-initializers-35aee1a28404

# Idea: *During Training,* Shift Activations So Resulting Gradients Support Learning Per Layer



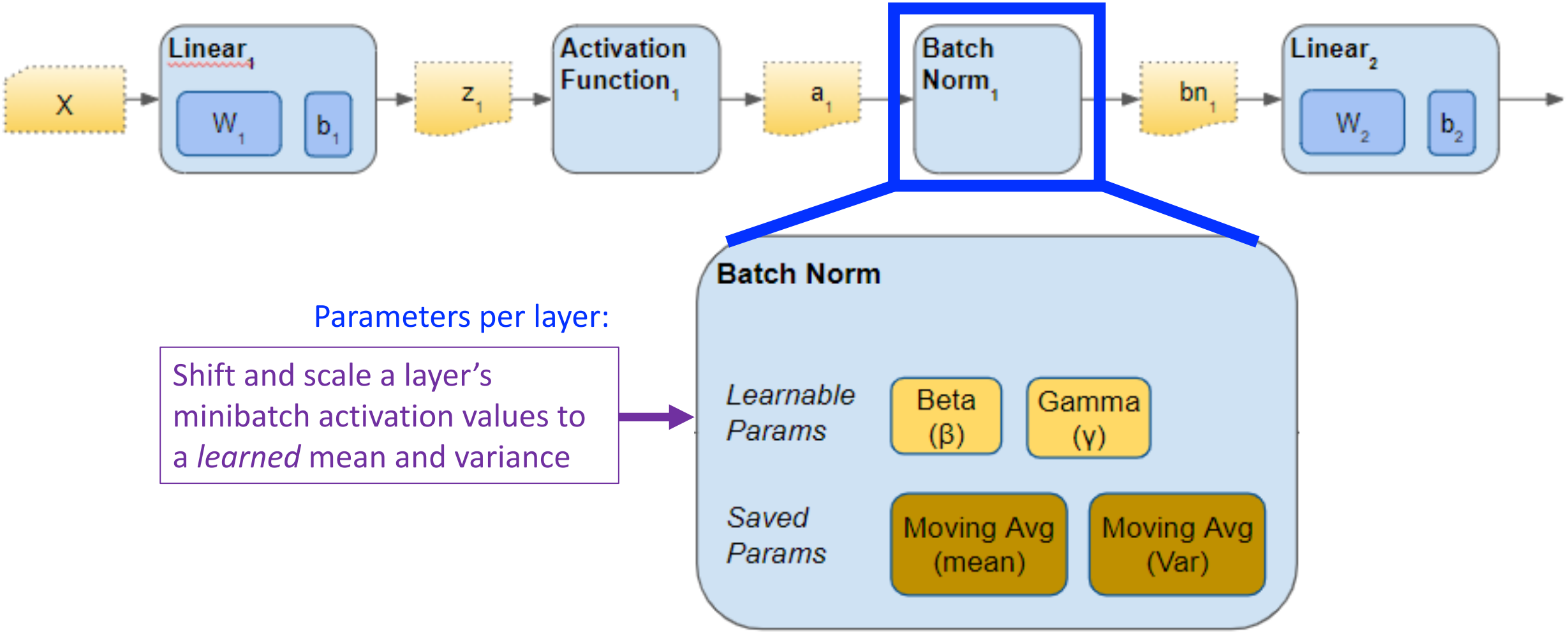

https://towardsdatascience.com/hyper-parameters-in-action-part-ii-weight-initializers-35aee1a28404

(2015, ICML)

# Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift

**Sergey Ioffe** SIOFFE@GOOGLE.COM
**Christian Szegedy** SZEGEDY@GOOGLE.COM
Google, 1600 Amphitheatre Pkwy, Mountain View, CA 94043

# Idea: *During Training,* Shift Activations So Resulting Gradients Support Learning Per Layer

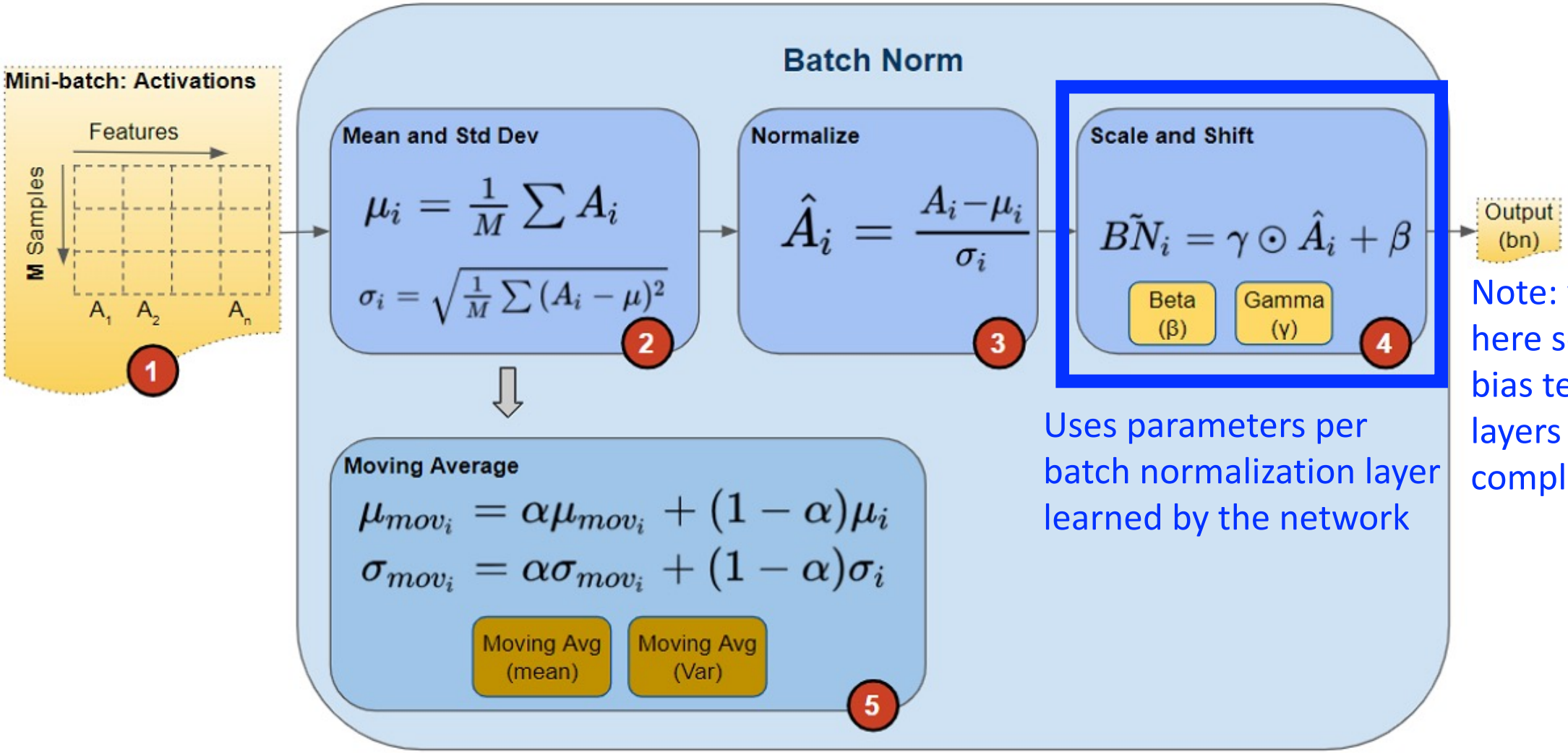



Parameters per layer:

Shift and scale a layer's minibatch activation values to a *learned* mean and variance

https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739

# Batch Normalization Layer: Training Operation



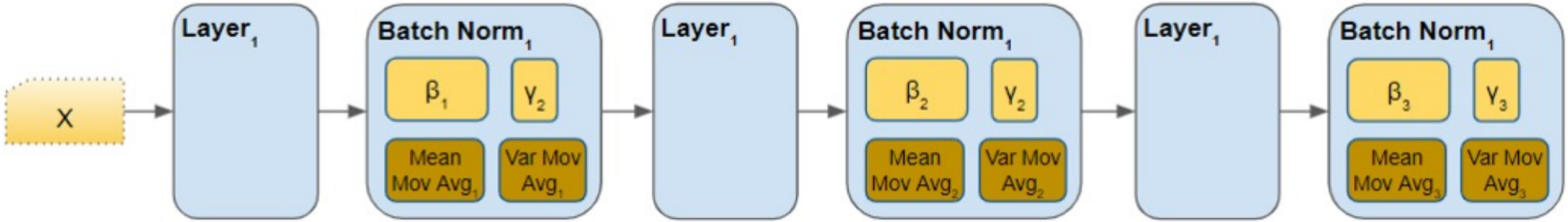

Uses parameters per batch normalization layer learned by the network

Note: we add a bias here so don't need bias terms in earlier layers (unnecessary complexity)

https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739

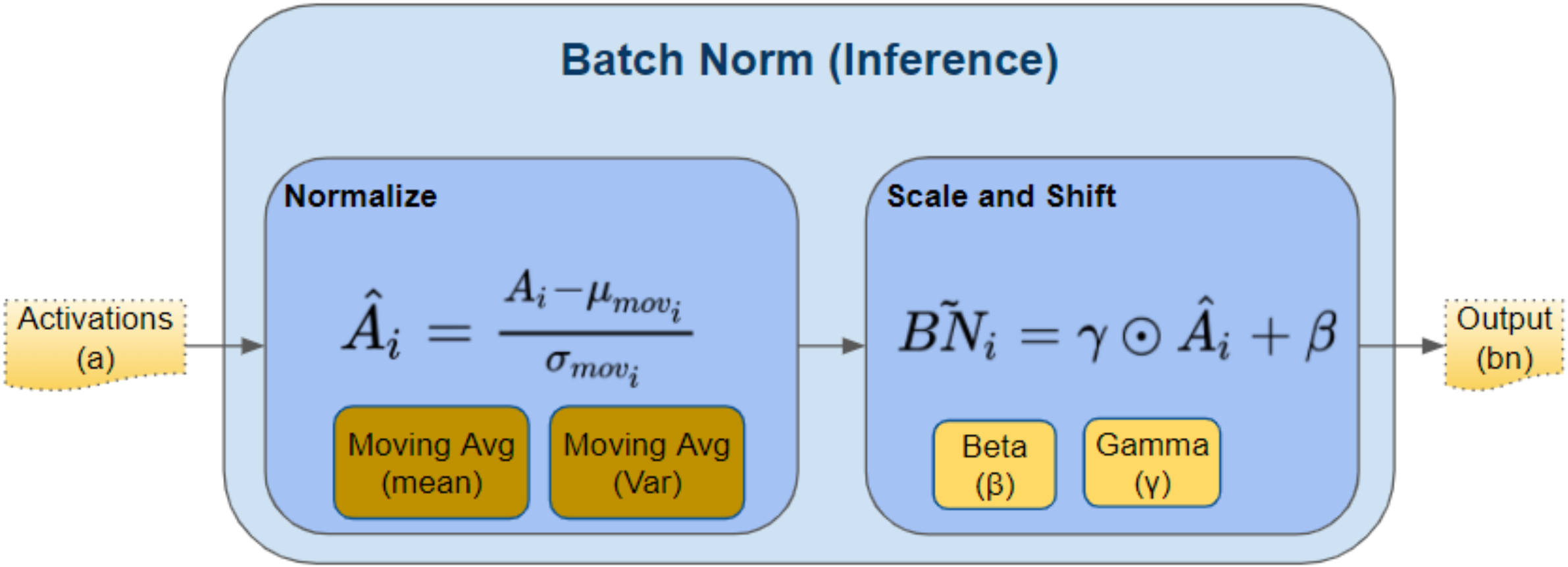
# Batch Normalization Layer: Training Operation

How many trainable parameters must be learned for batch normalization in this subnetwork?




https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739

# Batch Normalization Layer: Test-Time Operation




Layer brings a regularizing effect by introducing additive and multiplicative “noise”

https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739

# Benefits and Limitations

- Pros - smooths the optimization function leading to:
  - Faster training convergence
  - More stable learning when paired with different hyperparameters and initializations
  - Better generalization performance

- Cons
  - Extra layer(s) introduce more training and testing time
  - Examples in each mini-batch are coupled in forward pass (decoupled alternatives exist, such as instance normalization and group normalization)

Santurkar et al. How Does Batch Normalization Help Optimization? Neurips 2018.

# Today's Topics

- Regularization
- Parameter norm penalty
- Early stopping
- Dataset augmentation
- Dropout
- Batch normalization
- Programming tutorial

# Today’s Topics

- Regularization
- Parameter norm penalty
- Early stopping
- Dataset augmentation
- Dropout
- Batch normalization
- Programming tutorial

The End