

# Deep Learning with R

## Getting started

Mikhail Dozmorov

Virginia Commonwealth University

2020-06-08

## Welcome: Class format

Course web site: <https://bios691-deep-learning-r.netlify.app/>

- Lecture
- Live Coding
- Q&A Google Doc, <https://bit.ly/bios691>

You will get the most out of this class if you

1. Attend class
2. Follow and complete code examples yourself
3. Review references and reading
4. Ask questions

**Student Course evaluation, or Guest Course evaluation - please evaluate the course Friday afternoon, June 12, 2020. All evaluations are anonymous**

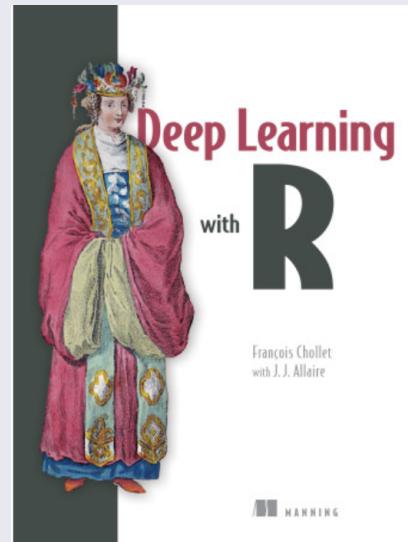
# Welcome: Textbook

Course web site: <https://bios691-deep-learning-r.netlify.app/>

Book: "Deep Learning with R" introduces the world of deep learning using the powerful Keras library and its R language interface

The book builds your understanding of deep learning through intuitive explanations and practical examples

<https://www.manning.com/books/deep-learning-with-r>



## Additional resources

- The Deep Learning textbook by Ian Goodfellow, Yoshua Bengio and Aaron Courville
- Deep Learning with Keras and TensorFlow in R Workflow - RStudio Conference 2020 workshop by Brad Boehmke
- MIT Introduction to Deep Learning | 6.S191 - MIT video course by Alexander Amini, Ava Soleimani, and guests. Dense and informative ~45min lectures covering various topics of deep learning.  
[introtodeeplearning.com](http://introtodeeplearning.com) - course web site with slides, video, and other material
- Machine learning and deep learning resources - a collection of references for further studies

# Final project

The objectives of this class include

- Become proficient in selecting the correct types of deep learning models
- Implement deep neural networks using Keras/TensorFlow and R programming language
- Train and evaluate the performance of deep neural networks

To help you with this, you will solve one of the **Kaggle competitions** based on topics/code learned in-class. See **Final project** for more details

- Teams of two are encouraged
- Due date is 07/12/2020
- Ask questions by e-mail

<https://www.kaggle.com/competitions>

6 / 42

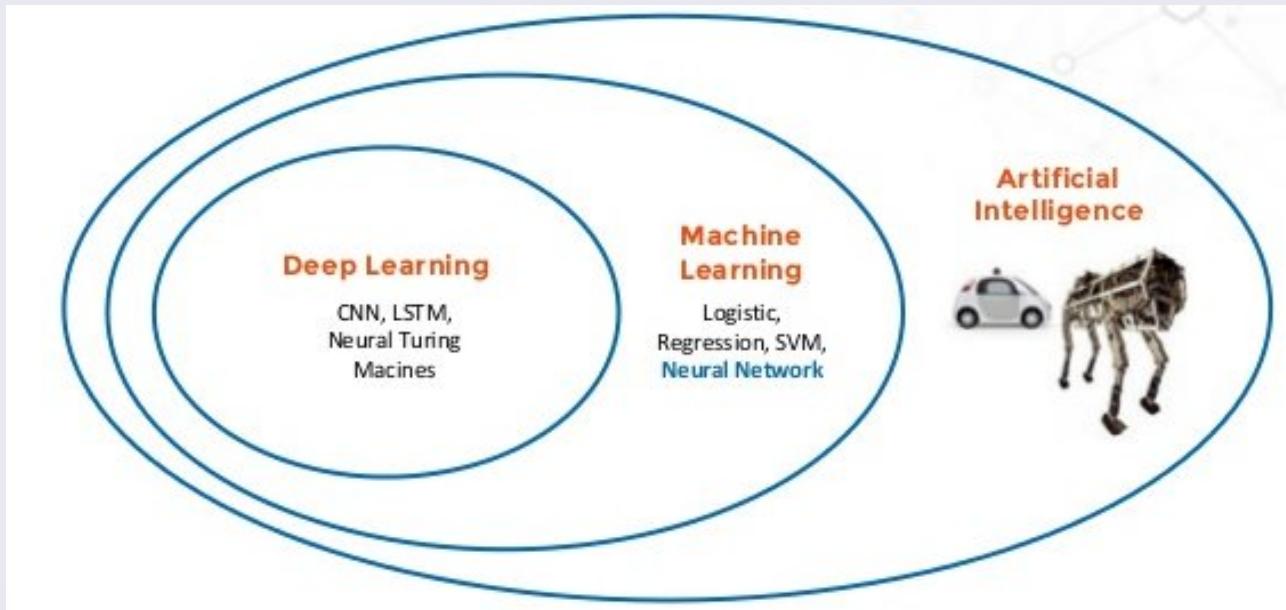
# Artificial Intelligence and Deep Learning

- **Artificial Intelligence** - A process of machines learning to perform tasks, rather than simply carrying out computations that are input by human users
- **Machine Learning** - An approach to AI in which a computer algorithm (a set of rules and procedures) is developed to analyze and make predictions from data that is fed into the system
- **Neural Networks** - A machine learning approach modeled after the brain in which algorithms process signals via interconnected nodes called artificial neurons
- **Deep Learning** - A form of machine learning that uses many layers of computation to form what is described as a deep neural network, capable of learning from large amounts of complex, unstructured data

<https://www.nibib.nih.gov/science-education/science-topics/artificial-intelligence-ai>

7 / 42

# Artificial Intelligence and Deep Learning

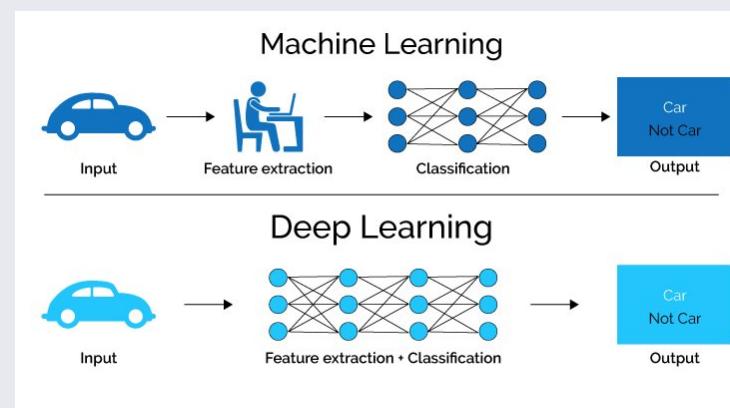


<https://leonardoaraujosantos.gitbooks.io/artificial-intelligence/content/chapter1.html>

8 / 42

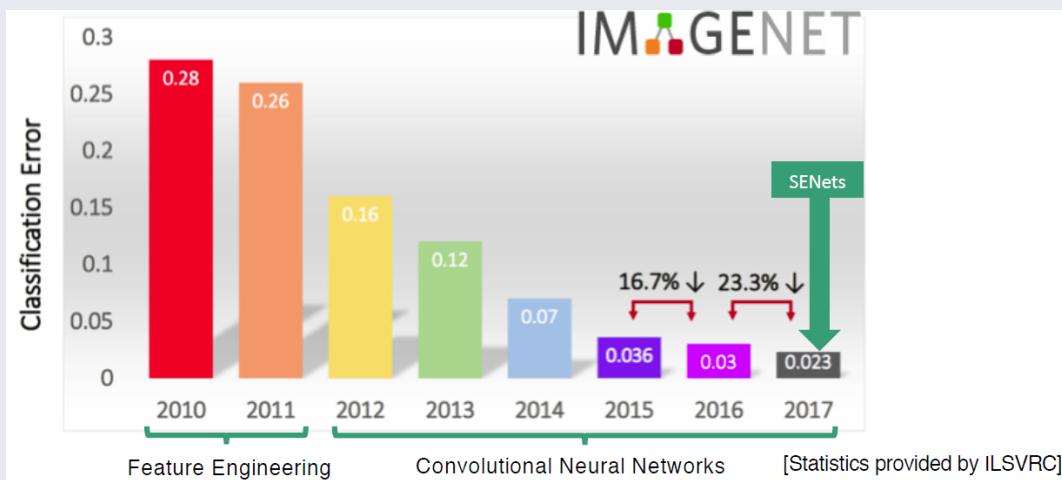
## Machine vs. Deep Learning

- **Feature engineering** - transforming input data so it is amenable for classification. Automated by Deep Learning
- **Feature selection** - selecting features most important for accurate classification



9 / 42

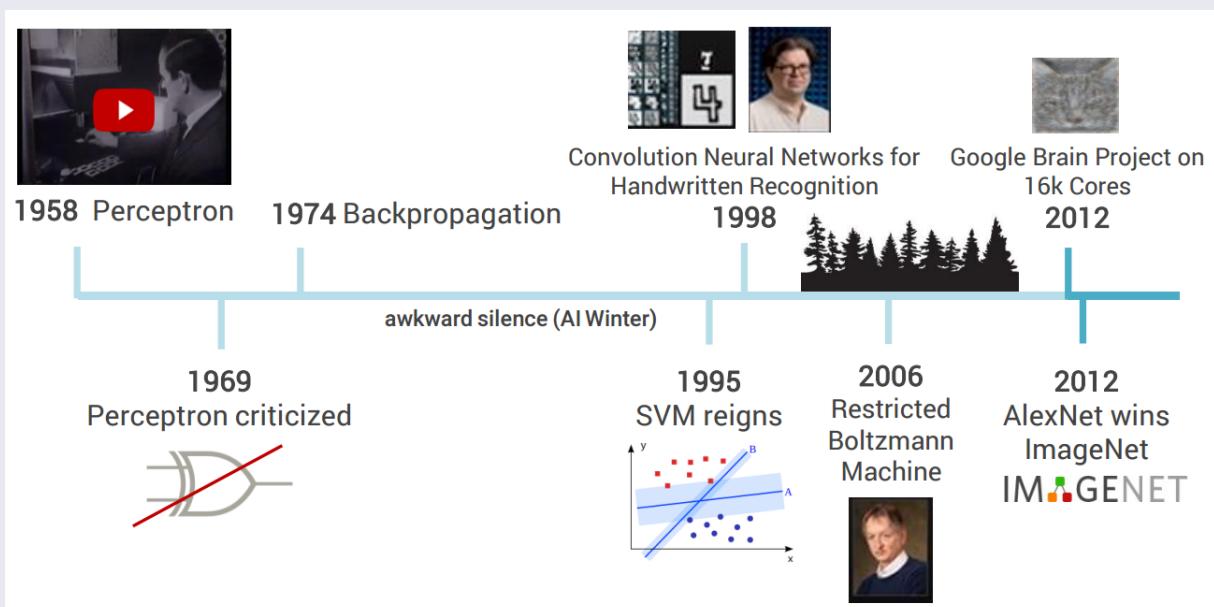
# Machine vs. Deep Learning



<https://towardsdatascience.com/review-senet-squeeze-and-excitation-network-winner-of-ilsvrc-2017-image-classification-a887b98b2883>

10 / 42

## Deep learning history



<https://leonardoaraujosantos.gitbooks.io/artificial-intelligence/chapter1.html>

11 / 42

# Deep learning applications

- Computer vision, image classification, image captioning
- Speech recognition, translation
- Text, natural-language processing, sentiment analysis
- Timeseries forecasting
- Image, text, speech/music generation, music-to-notes translation
- Self-driving cars

<http://detexify.kirelabs.org/classify.html>

12 / 42

# Essential Knowledge for Deep Learning

- Probability and Statistics
- Linear Algebra and Calculus
- Key Machine Learning Concepts
- Hardware (GPU preferred)
- Software (Keras preferred)
- Programming (Linux-based OS preferred, Python preferred)

<https://www.analyticsvidhya.com/blog/2020/03/deep-learning-5-things-to-know/>

13 / 42

# Probability and Statistics for Deep Learning

- **Descriptive Statistics** - Standard Deviation, Variance, Normal distribution, Central Limit Theorem
- **Probability** - random variables, binomial distribution, Z-scores, significance level
- **Linear models** - easily interpretable machine learning

14 / 42

# Linear Algebra for Deep Learning

- **Scalars and vectors**
  - **Dot product** - the Dot product of two vectors returns a scalar value
  - **Cross product** - the Cross product of two vectors returns another vector which is orthogonal (right-angled) to both
- **Matrices and Matrix Operations**
  - **Scalar Multiplication** - multiply all matrix elements with the scalar
  - **Matrix Multiplication** - multiplying two matrices means calculating the dot product of the rows and columns and creating a new matrix with dimensions derived from the two input matrices
  - **Hadamard product** - element-wise product of the two vectors (  $\odot$  )
  - **Transpose of the matrix** - swap the rows and the columns in a matrix
  - **Inverse to the matrix** - an inverse of a matrix multiplied with the matrix gives an identity matrix

15 / 42

# Tensors

Tensors are a generalization of vectors and matrices to an arbitrary number of dimensions

- A tensor that contains only one number is called a scalar (or scalar-tensor, or zero-dimensional tensor, or 0D tensor)
- A one-dimensional array of numbers is called a vector, or 1D tensor
- A two-dimensional array of numbers is a matrix, or 2D tensor
- An array of matrices makes a three-dimensional array of numbers, or 3D tensor

16 / 42

# Tensors

A tensor is defined by three key attributes:

- **Number of axes (rank)** - For instance, a 3D tensor has three axes, and a matrix has two axes
- **Shape** - This is an integer vector that describes how many dimensions the tensor has along each axis. A vector has a shape with a single element, such as (5). Think `dim()` function.
- **Data type** - This is the type of the data contained in the tensor; integer or double

17 / 42

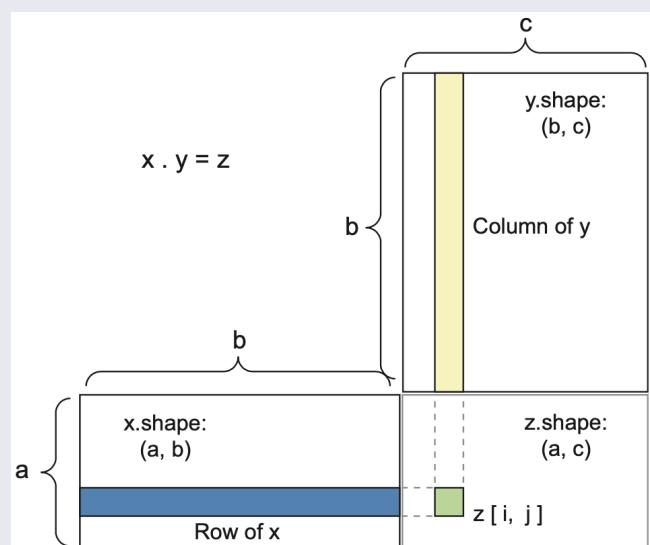
# Real-world tensors

- **Vector data** - 2D tensors of shape (samples, features)
- **Timeseries data or sequence data** - 3D tensors of shape (samples, timesteps, features)
- **Images** - 4D tensors of shape (samples, height, width, channels)
- **Video** - 5D tensors of shape (samples, frames, height, width, channels)

18 / 42

## Tensor operations

- Addition, multiplication
- Tensor transformation with functions
- Tensor reshaping
- Tensor dot<sup>1</sup>



review Section "2.3.3 Tensor dot", p. 36

19 / 42

# Calculus for Deep Learning

- **Derivatives** - derivatives measure the change in the output value when we change the input value
- **Partial derivatives** - Partial derivative is when we consider only one variable, keep all the other variables as constant, and take the derivative with respect to the considered variable
- **Chain rule** - helps when input variables are expressed as functions of other functions. If  $y = f(u)$  and  $u = g(x)$  are both differentiable, then
$$\frac{dy}{dx} = \frac{dy}{du} \frac{du}{dx}$$
- **Gradient** - the gradient of the function  $f(x)$  with respect to  $x$  (an  $n$ -dimensional vector  $[x_1, x_2, \dots, x_n]^T$ ) is a vector of  $n$  partial derivatives
$$\nabla_x f(x) = \left[ \frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \dots, \frac{\partial f(x)}{\partial x_n} \right]^T$$

[https://d2l.ai/chapter\\_preliminaries/calculus.html](https://d2l.ai/chapter_preliminaries/calculus.html)

20 / 42

# Supervised Learning

- **Supervised Learning** - using training data (input variables) with the known target variable, find rules that give the relationship between the input and the target variables. These rules can then be applied to the unseen (test) data to predict the expected target variables for it.  
Examples: kNN, SVM, Linear Regression, etc.
  - **Regression** - predict numeric (continuous) outcome
  - **Classification** - predict continuous (binary or multinomial) outcome
- **Applications** - Text categorization, Face Detection, Signature recognition, Customer recommendation system, Spam detection, Weather forecasting, Predicting housing prices based on the prevailing market price, Stock price predictions, among others

<https://www.analyticsvidhya.com/blog/2020/04/supervised-learning-unsupervised-learning/>

21 / 42

# Unsupervised Learning

- **Unsupervised Learning** - the target variable is unknown, so the goal is to cluster the data into groups, and we can identify the groups after we have clustered the data. Examples of unsupervised learning include k-means clustering, dimensionality reduction techniques, etc.
  - **Clustering** - organize unlabelled data into *similar* groups
  - **Anomaly detection** - identify rare events
- **Applications** - Fraud detection, Malware detection, Identification of human errors during data entry, Conducting accurate basket analysis, etc.

<https://www.analyticsvidhya.com/blog/2020/04/supervised-learning-unsupervised-learning/>

22 / 42

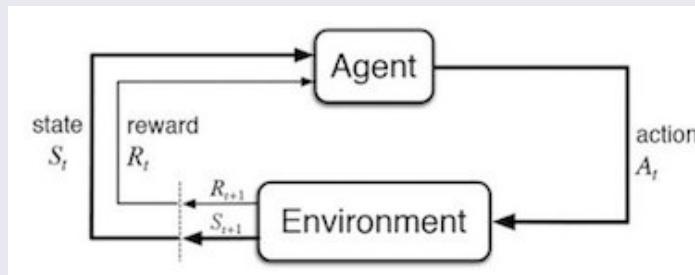
# Semi-supervised learning

- Learning guided by the data itself
- Labels generated from the input data
- **Autoencoders** - learns data properties (labels) to reconstruct input data
- The distinction from other learning types is not well-defined

23 / 42

# Reinforcement Learning

- **Reinforcement Learning** - learning to make specific decisions based on past experience. An agent interacts with an environment over time, receives feedback as reward or punishment, and adjusts its action policy to maximize the reward for the next action.
  - 2015: Google's Deep Q algorithm wins Atari games
  - 2015: Google's AlphaGo algorithm beats best human Go players

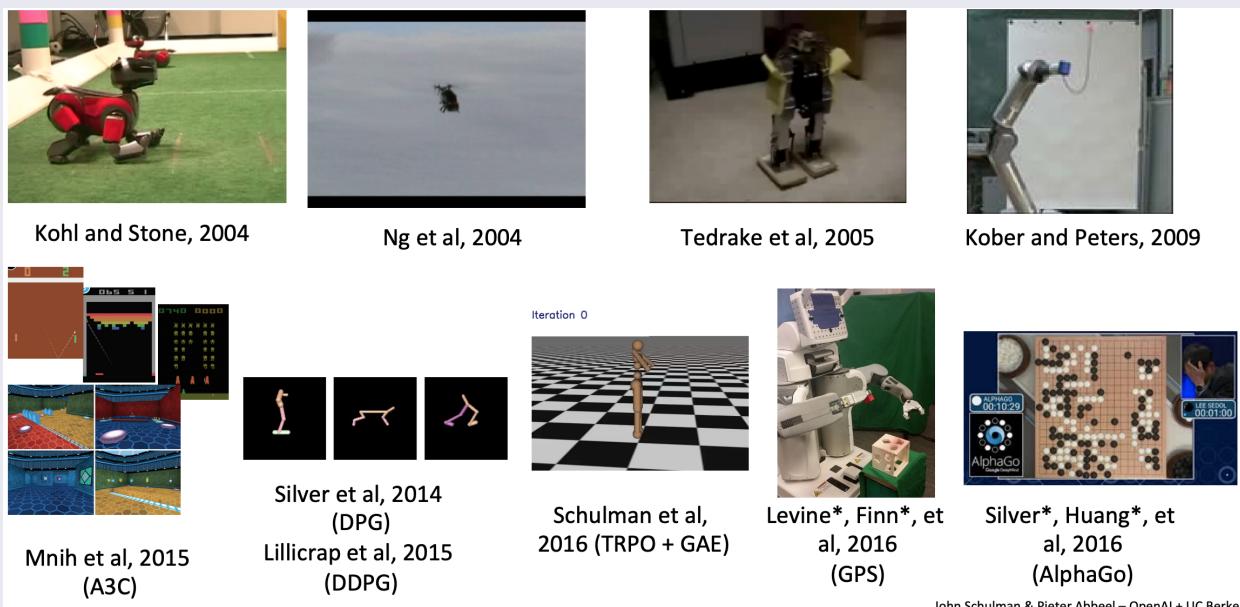


<https://www.wired.com/2015/02/google-ai-plays-atari-like-pros/>

<https://www.wired.com/2017/05/googles-alphago-trounces-humans-also-gives-boost/>

24 / 42

# Reinforcement Learning



<https://simons.berkeley.edu/sites/default/files/docs/6453/201703xxsimons-representations-deep-rl.pdf>

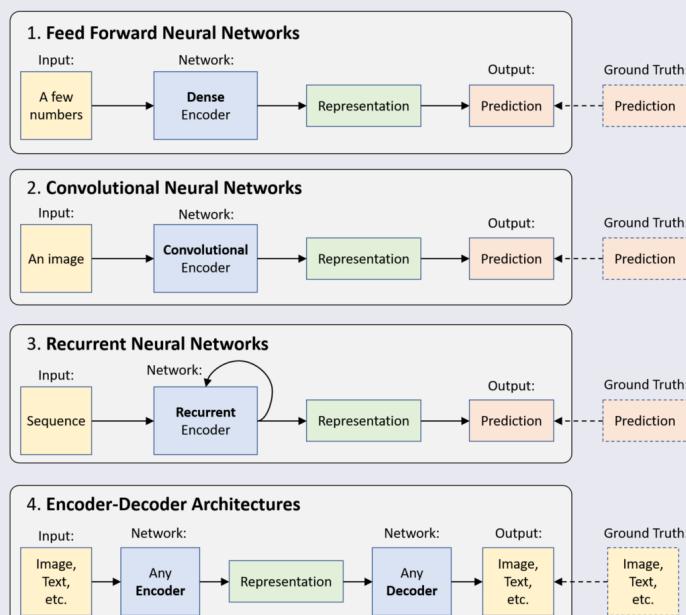
25 / 42

# Transfer learning

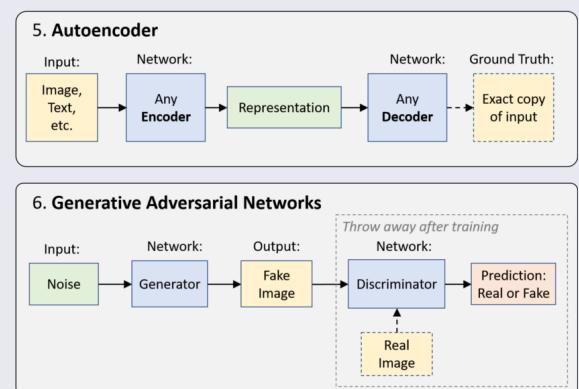
- Train a large model on a huge dataset, and then use the final parameters on smaller datasets
- Pretrained models significantly simplify and reduce time of training
- Popular in computer vision ( VGG-16, ResNet ), NLP (GPT-2, BERT)

26 / 42

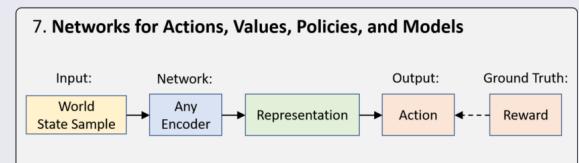
## Supervised Learning



## Unsupervised Learning



## Reinforcement Learning



<https://medium.com/tensorflow/mit-deep-learning-basics-introduction-and-overview-with-tensorflow-355bcd26baf0>

27 / 42

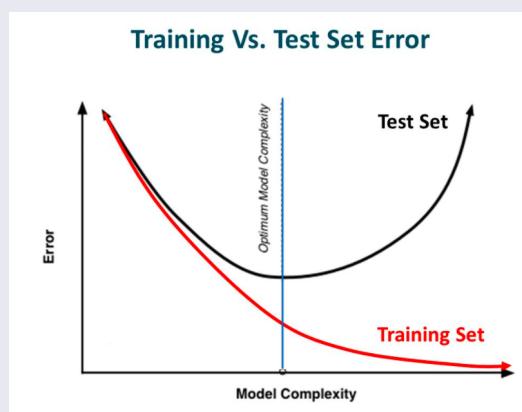
# Data preprocessing

- **Vectorization** - sample-specific data should be represented as a vector. The vector representation scheme should be consistent from sample to sample
- **Normalization** - Make values small, typically within the 0-1 range. Make them homogeneous, relatively uniformly distributed throughout the range
  - In general, it isn't safe to feed into a neural network data that takes relatively large values or data that are on different scales. Doing so can trigger large gradient updates that will prevent the network from converging
  - Normalize **features** to have a mean of 0 and a standard deviation of 1
  - Important: Normalize training data separately, then use these normalization constants to normalize validation and test sets

28 / 42

# Training-test-validation sets

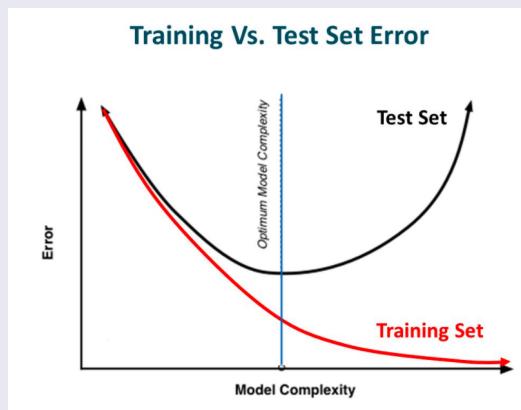
- **Training set** - data used to develop feature sets, train our algorithms, tune hyperparameters, compare models, and all of the other activities required to choose a final model
- **Test set** - data used to estimate the final model's performance
- 70%/30% split is typical to separate the initial dataset into training and test sets



29 / 42

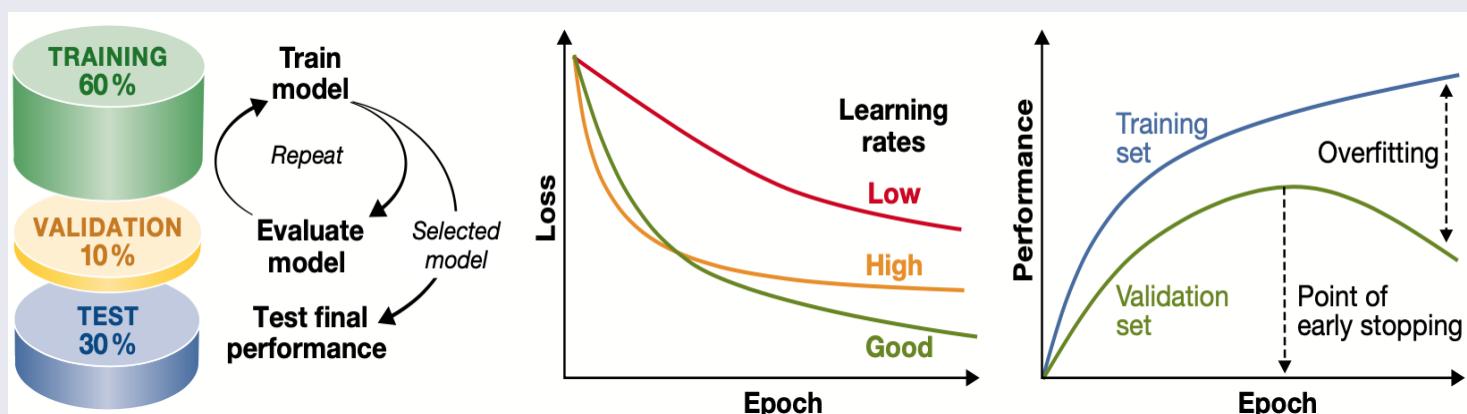
# Training-test-validation sets

- During training, some data leakage occurs - the model memorizes some properties of the test set that improve performance. Leads to overfitting
- **Validation** - data used to improve the model. Needed to evaluate the model on never-before-seen data, to make the model generalizable and prevent overfitting
- 60%/20%/20% is typical to split the data into three sets



30 / 42

# Training-test-validation sets

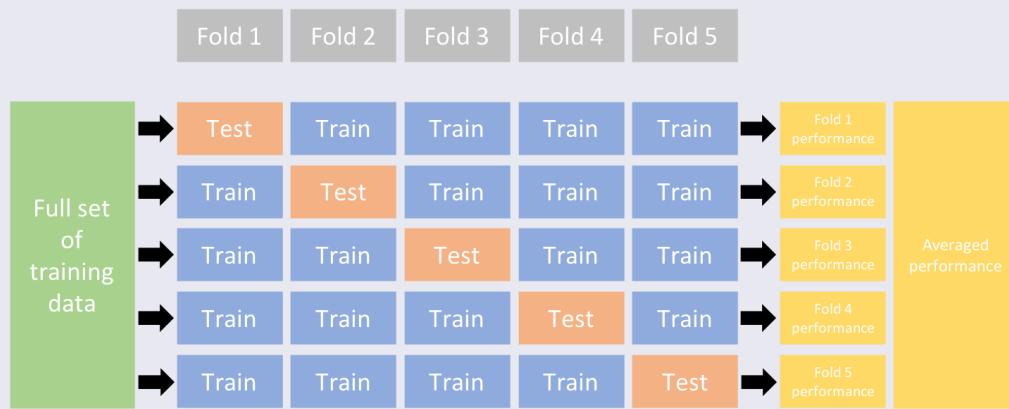


Angermueller et al., "Deep Learning for Computational Biology."

31 / 42

# Cross-validation

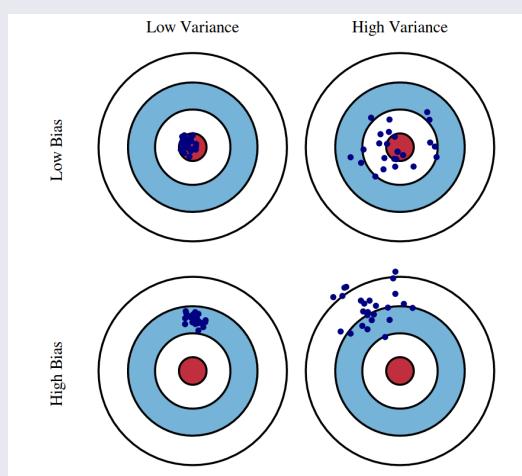
- **Cross-validation** - A common machine learning strategy wherein the dataset is split multiple times into training and validation sets. The average validation performance across the multiple splits is used to select the final model
- k-fold cross-validation, leave-one-out cross-validation, bootstrapping



32 / 42

## Bias variance trade-off

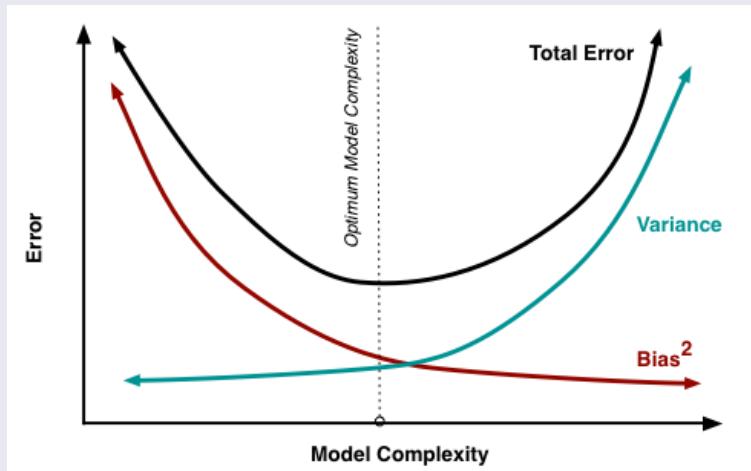
- **Bias** - the difference between the expected (or average) prediction of our model and the correct value which we are trying to predict
- **Variance** - the variability of a model prediction for a given data point



33 / 42

# Bias variance trade-off

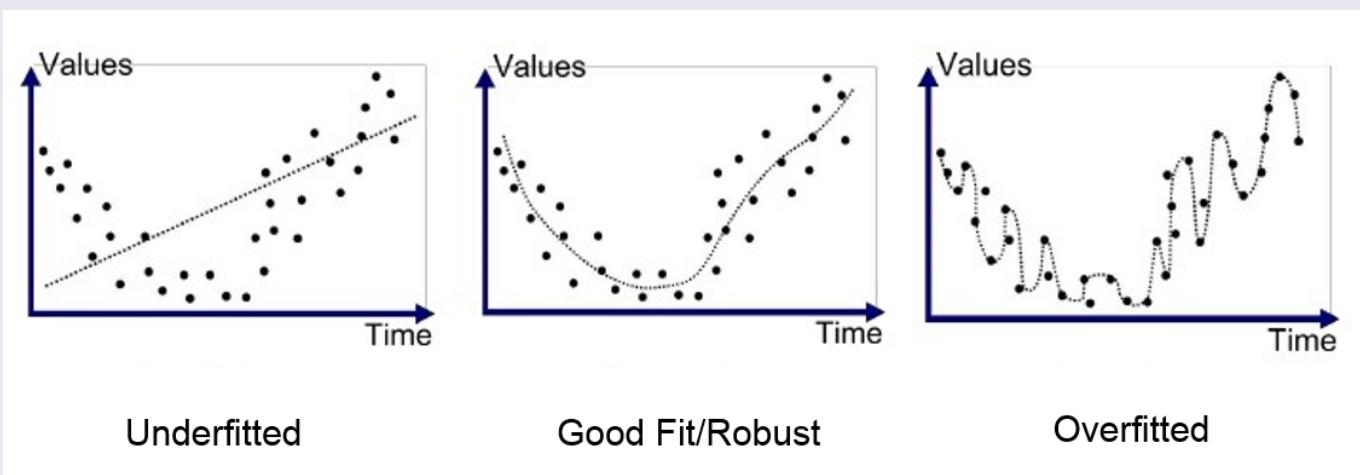
- **Bias** - the difference between the expected (or average) prediction of our model and the correct value which we are trying to predict
- **Variance** - the variability of a model prediction for a given data point



34 / 42

## Underfitting and Overfitting

- **Underfitting and Overfitting** - too simple or too complex models



<https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76>

35 / 42

# Architectures

- Deep Learning methods comprise a wide variety of architectures. The most popular ones are
  - Multilayer Perceptron (MLP), fully connected networks
  - Convolutional Neural Networks (CNN)
  - Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Units (GRU)
  - Generative Adversarial Networks (GAN), (Variational) Autoencoders (VAE)

<https://www.analyticsvidhya.com/blog/2020/02/cnn-vs-rnn-vs-mlp-analyzing-3-types-of-neural-networks-in-deep-learning/>

36 / 42

# Hardware

- GPU, Graphics Processing Unit - highly parallel computations, speeds up network training
  - NVIDIA K80, P4, P100 - starter GPUs, on the order of increased performance
  - NVIDIA GTX1060 and above - intermediate GPUs
  - NVIDIA V100 - advanced GPU, up to 47X speedup compared with CPU
- TPU, Tensor Processing Unit - designed for parallel tensor computations, faster and cheaper than GPUs

<https://towardsdatascience.com/maximize-your-gpu-dollars-a9133f4e546a>

37 / 42

# Software

- **TensorFlow** - the most popular framework, developed by Google. Has an R interface
- **Keras** - the front-end API to the most popular frameworks (Tensorflow, Theano, others). Much less and simpler code. Starting point for beginners
- **PyTorch** - the distinct framework, highly customizable, gaining popularity
- **FastAI** - founded by Jeremy Howard, the former President and Chief Scientist at Kaggle. PyTorch-based well-tuned components for best performance. Free online courses

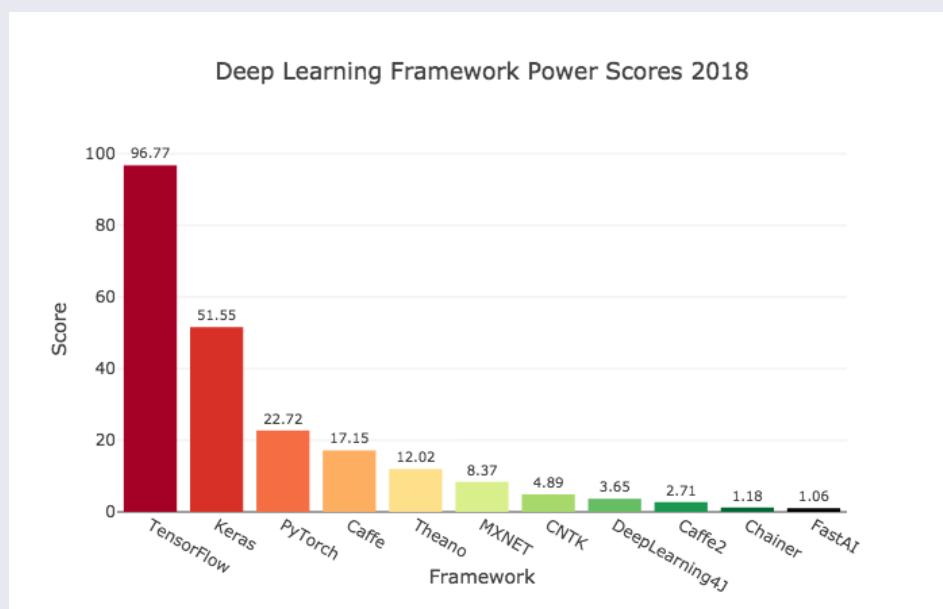
<https://tensorflow.rstudio.com/>, <https://keras.rstudio.com/>

[https://tensorflow.rstudio.com/installation/gpu/local\\_gpu/](https://tensorflow.rstudio.com/installation/gpu/local_gpu/)

<https://www.analyticsvidhya.com/blog/2020/03/tensorflow-2-tutorial-deep-learning/>

38 / 42

# Software



<https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a>

39 / 42

# Providers

- **Google Colab** - free Jupyter notebooks, integrated with GitHub and Google Drive, provides free entry-level GPU and TPU
- **Google Cloud** - powerful cloud computing, paid, easy to navigate interface and price plans. Preemptible (short-lived but affordable) virtual machines
- **AWS EC2** - popular cloud computing platform, requires understanding of pricing options. \$300 credit for starters
- **Kaggle** - free Jupyter notebooks, with GPUs. Lots of community code
- Other options: **Paperspace**, **vast.ai**, **Microsoft Azure**

<https://www.analyticsvidhya.com/blog/2020/03/google-colab-machine-learning-deep-learning/>

<https://towardsdatascience.com/maximize-your-gpu-dollars-a9133f4e546a>

40 / 42

## How to think about deep learning

- The most surprising thing about deep learning is how simple it is
- All you need is sufficiently large parametric models trained with gradient descent on sufficiently many examples
- Each layer in a deep-learning model operates one simple geometric transformation on the data that goes through it
- A key characteristic of this geometric transformation is that it must be differentiable, which is required in order for us to be able to learn its parameters via gradient descent. Intuitively, this means the geometric morphing from inputs to outputs must be smooth and continuous
- As Richard Feynman once said about the universe, "It's not complicated, it's just a lot of it"

41 / 42

# Leaders of Deep Learning

*backpropagation,  
boltzmann machines*



Geoff Hinton  
Google

*convolution*



Yann Lecun  
Facebook

*stacked auto-  
encoders*



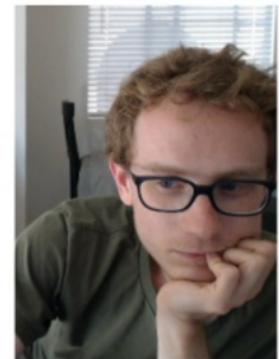
Yoshua Bengio  
U. of Montreal

*GPU utilization*



Andrew Ng  
Baidu

*dropout*



Alex Krizhevsky  
Google

- Heroes of Deep Learning, Interviews by Andrew Ng

[https://leonardoaraujosantos.gitbooks.io/artificial-intelligence/deep\\_learning.html](https://leonardoaraujosantos.gitbooks.io/artificial-intelligence/deep_learning.html)

42 / 42

## Welcome: Textbook

François Chollet: Keras, Deep Learning, and the Prog...



TensorFlow and Keras in R - Josh Gordon meets with...



Authors: François Chollet (the creator of Keras) with J. J. Allaire (the founder of RStudio and the author of the R interfaces to Keras and TensorFlow)

<https://github.com/jjallaire/deep-learning-with-r-notebooks>

4 / 42