alitourani

Deep Learning from Scratch

Session #3: Feeding DNNs

by: Ali Tourani – Summer 2021

Agenda

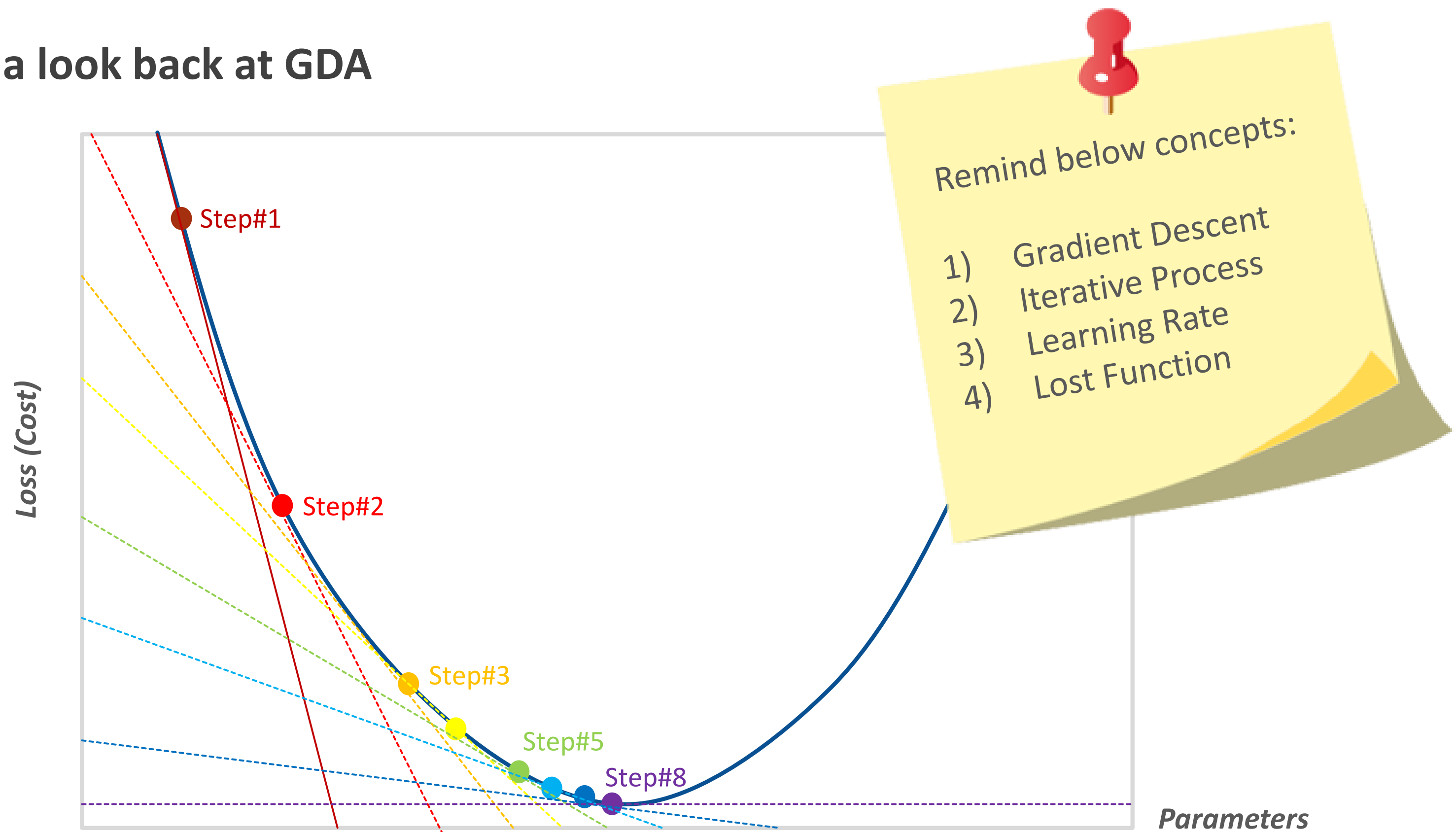
- Warm-up and Review
- Essential Concepts
- Hyperparameters
- Overfitting Problem
- Learning Paradigms
- Roadmap

Warm-up and Review

- Types of data: image, video, sound, text, time series
- Datasets
- Standard vs. real-world data
- Generate datasets
- Data labeling/annotation
- Where to find data?
  - Google's dataset search engine, Kaggle, etc.
- Deep learning and data
  - Training-set, dev-set, test-set

Essential Concepts

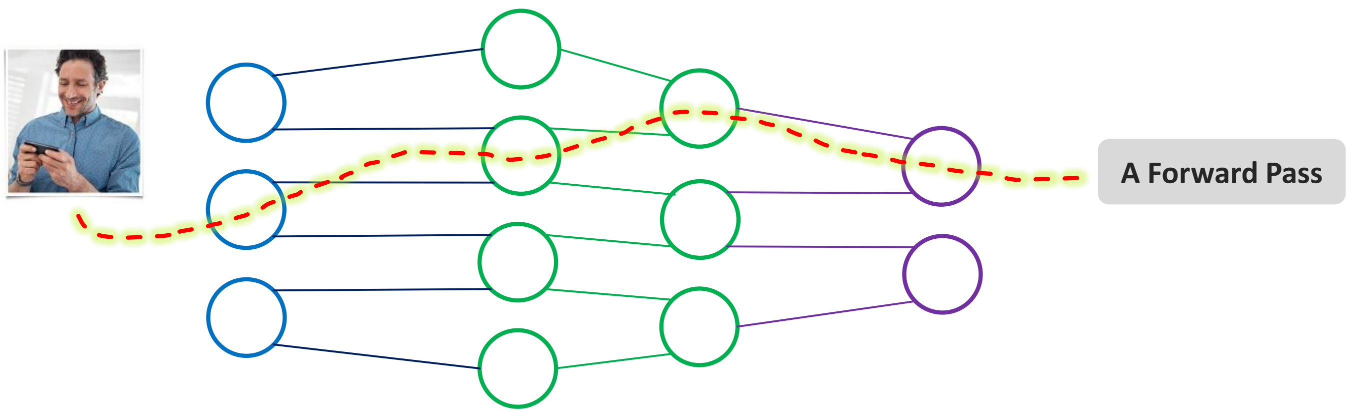
Take a look back at GDA



Essential Concepts

I. Pass

- A **forward pass (calculation)** or a **backward pass (learning)** in an ANN
- Traversing through all neurons of a neural network
- It might be time-consuming, considering the number of hidden layers



Essential Concepts

II. Batch

- A subset of the training-set, AKA a ***Mini-Batch***
  - **Goal:** feeding the NN with a limited number of instances <u>iteratively</u>
- The number of data used in one forward/backward pass
- **Batch size:** the number of cases collected from the training-set

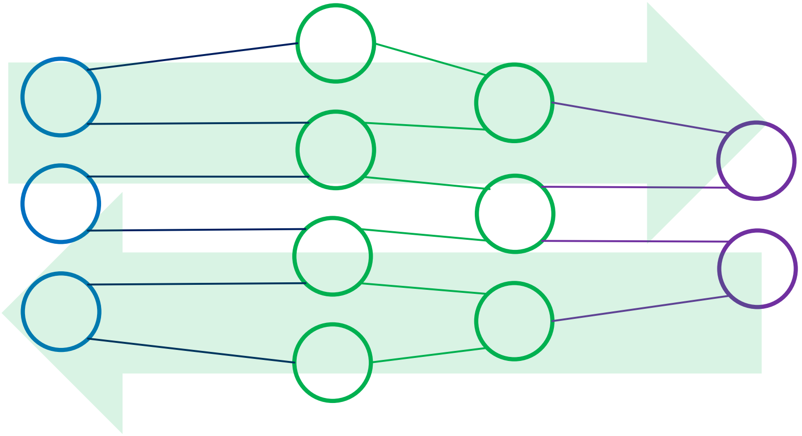
Training data

Batches Size = 3

Essential Concepts

III. Iteration

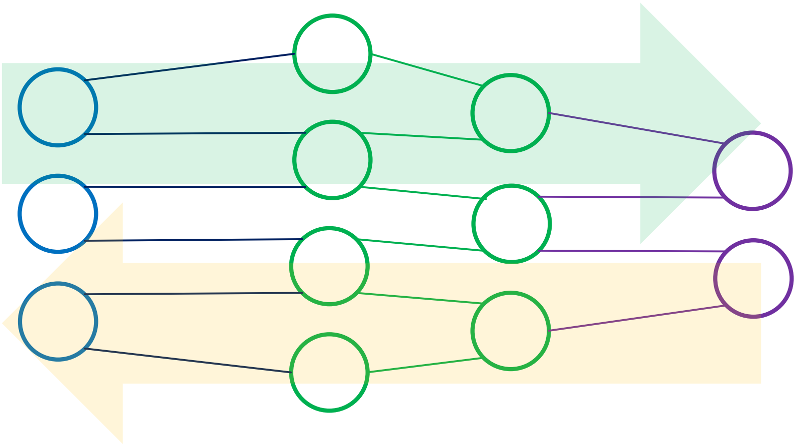
- A backward and forward pass of a batch of data
- *The number of iterations?*
  - Number of passes, each using a ***[batch size]*** number of instances
- We may need calculations for this!



Essential Concepts

IV. Epoch

- Passing the entire dataset **forward and backward** ONLY ONCE
- The number of times the algorithm works through the whole training-set
- One epoch contains $dataset Size / batch Size$ iterations



Essential Concepts

Important Notes

- Batch size and number of batches are not equal
- Setting a batch size is essential, as we cannot pass the whole training set into the ANN at once
- The final batch may contain fewer samples than the other batches
- The whole training set should be passed to the ANN multiple times
- The more number of epochs, the more learning processes
- There is no magic rule for choosing the right number of epochs and batch size

Essential Concepts

Let's see an example

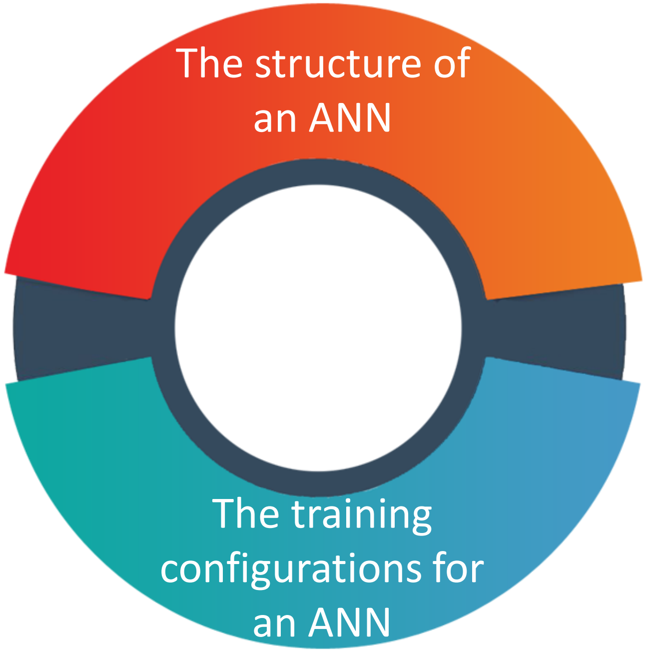
- **10,000 images** of human face (training data)
- *Batch-size*: **500**
- *# of iterations*: **20**

So, if we divide a dataset of 10,000 samples into batches of size 500, we will have to wait for 20 iterations to complete one epoch.

Image source: https://www.arxiv.org

Hyperparameters

- Variables and parameters to define:



It is so important to set the hyperparameters **before training the network** (one of the main factors to gain magnificent results and accurate predictions).

Hyperparameters

I. Network Structure-related Hyperparameters

Hyperparameter	How to tune?	Notes
Number of Hidden Layers	Adding hidden layers until the error does not improve in the test	Having too many layers is as inefficient as too low layers
Network Weight Initialization	Using different weight initialization schemes according to the AFs	Uniform distribution might be a good idea in many cases
Activation Functions	Choosing proper AFs according to their functionality and usage	Using a particular type of AF on all layers will result in improper outcomes

Hyperparameters

II. Training Algorithm-related Hyperparameters

Hyperparameter	How to tune?	Notes
Learning Rate	Considering the effects of low and large LRs in the ANN's performance	Decaying or Adaptive LRs are usually preferred
Batch Size	Trying 32, 64, 128, 256, and so on	-
Number of Epochs	Increasing the number of epochs and checking the validation accuracy	-

Hyperparameters

Important Notes

- Hyperparameters values are used to control the overall learning process
- They cannot be directly trained from the data
  - So, Hyperparameters are not model parameters
- Model parameters *(e.g. weights, coefficients, etc.)* are fetched from data, and hyperparameters are manually used to estimate their values
- Tuning of HPs lead to minimized loss functions and optimized models
- A good practice for tuning them can be as follows:

Model Definition → **Sampling possible hyperparameter values** → **Evaluation & cross-validation**

Hyperparameters

How to tune them?

- We can simply build a model for each possible combination of the HPs
- **Grid Search**
  - Iterates on all different permutations of values to select the most appropriate one
  - Might be inefficient in some scenarios

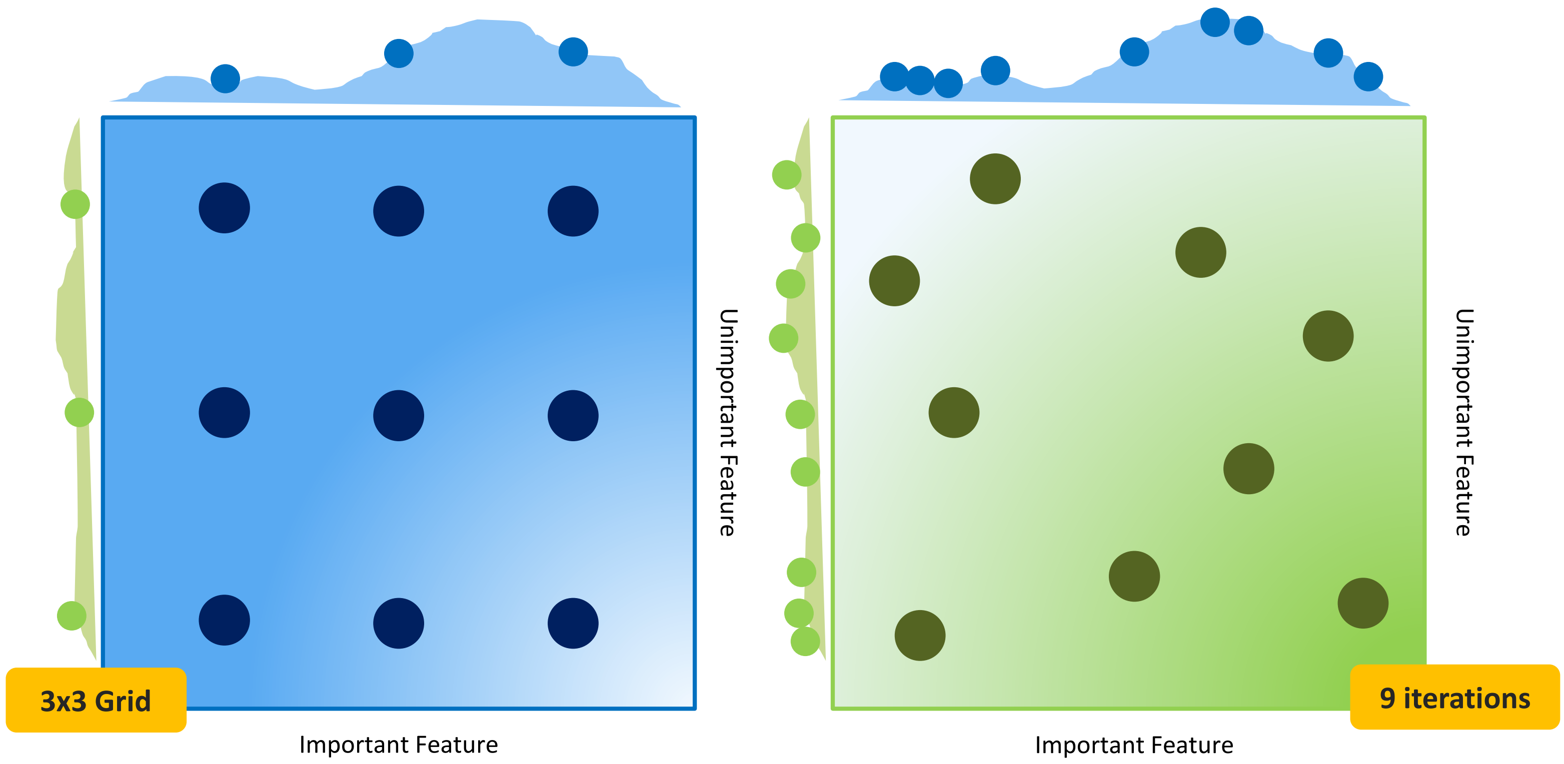
$HP1 = [1, 2, 3, 4, 5]$ $\quad HP2 = [0, 9, 18, 27, 35]$ $\quad Result = \{(1,0), (1,9) \ldots (5, 35)\}$

- **Random Search**
  - Providing an statistical distribution for each hyperparameter (random sampling)

$HP1 = random(1, 5)$ $\quad HP2 = random(0, 35)$

Hyperparameters

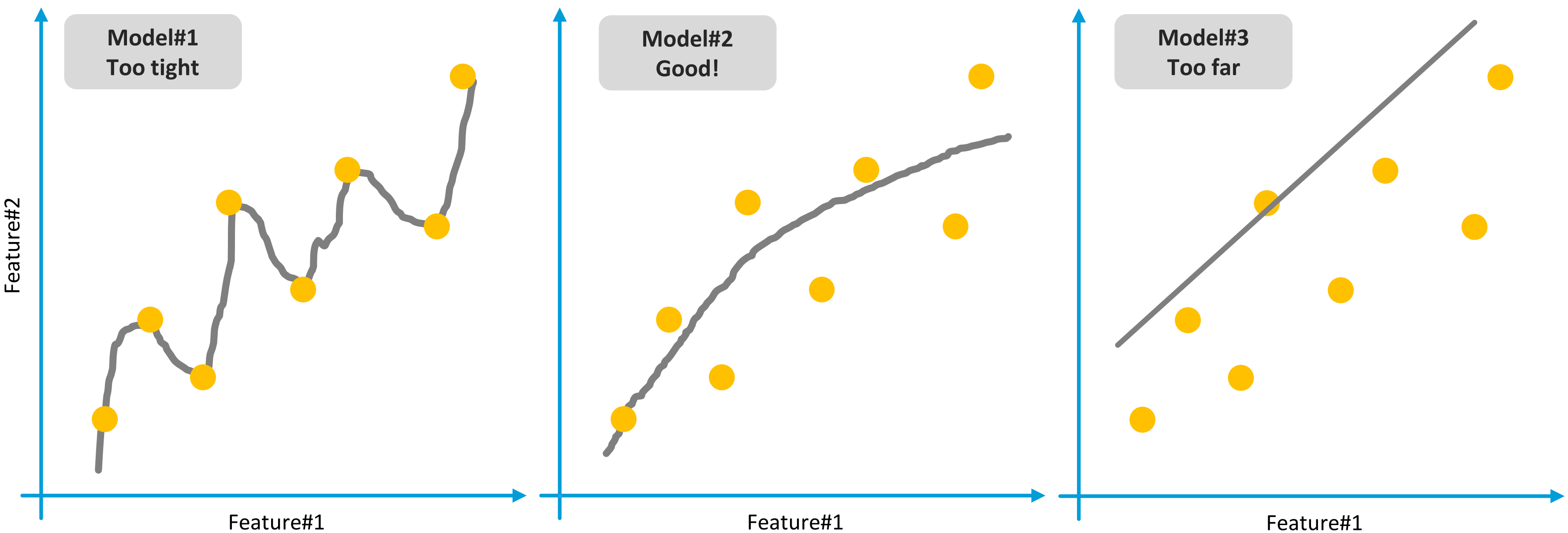
Grid Search vs. Random Search



Overfitting Problem

Consider the DNN model below:

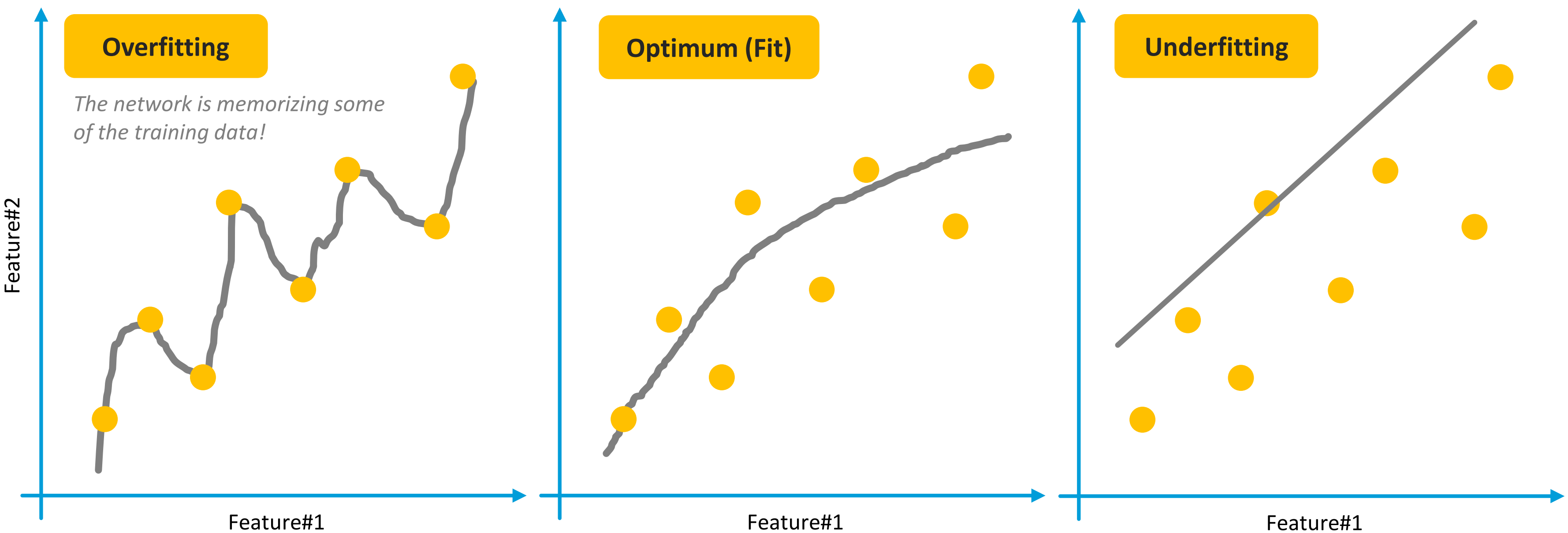
- **Goal:** finding a model that fits data samples with an acceptable error



Overfitting Problem

Consider the DNN model below:

- **Goal:** finding a model that fits data samples with an acceptable error

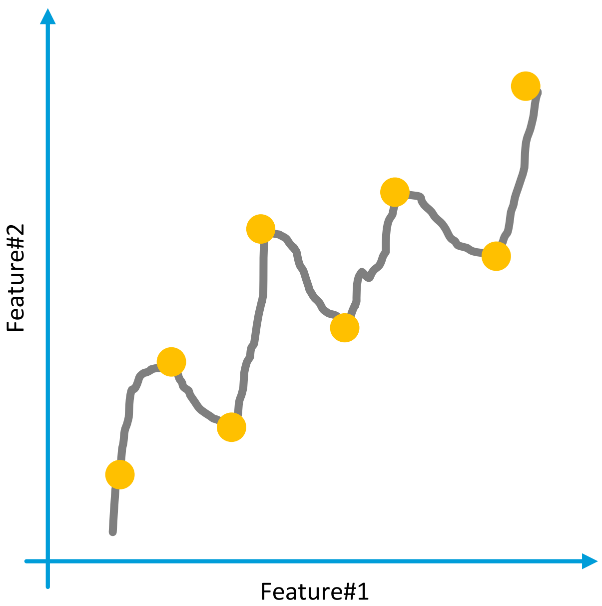


Overfitting Problem

The Concept of Overfitting

- Generalization Problem in ML?
  - The model is not trained to solve **General Problems**
  - Accurate predictions on the training set but inaccurate on unseen data
  - A complex model containing extra parameters and data
- Detecting the relationships in training data that

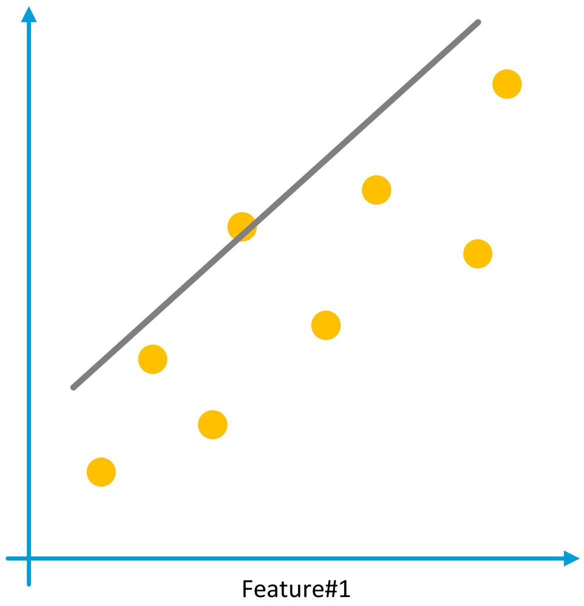
  are not held in **GENERAL**
- Fitting the line instead of finding the trend



Overfitting Problem

The Concept of Underfitting

- The model cannot provide accurate predictions even on the training set
- Unable to find a capacity to learn data fully
- The model could not:
  - Learn enough patterns from the training data
  - Capture the dominant trend
- Might be called ***"high bias"*** in some cases



Overfitting Problem

Important Notes

- The primary purpose of all ML models is to generalize well
- As we want the error to become smaller with more iterations, the process of overfitting detection should be based on the errors
- We should always try to find the trend instead of fitting the line
- Training with one epoch will lead to an underfitted model
- The training flow changes as: underfitted → optimum → overfitted
- Underfitting is as bad for the generalization of the model as overfitting

Overfitting Problem

How to avoid Overfitting?

- *Method#1: Training the ANN with more data*
  - Reducing the capacity of the model to overfit a training set
  - Training the model with more data will increase the generalization
  - **Data augmentation** is a similar approach
- *Method#2: Multiple Neural Networks*
  - Running several ANNs in parallel on the same training set
  - Using different initial weights and configurations
  - Comparing their error with the error of their average

Overfitting Problem

How to avoid Overfitting?

- *Method#3: Constraining Model Complexity*
  - Decreasing the complexity of the model and making the ANN smaller
    - Removing layers and reducing the number of neurons
  - Avoiding the network to catch all data points
  - Pruning it by removing nodes until it achieves a suitable performance
  - Any *magic rule* for the amount of simplification? Unfortunately not!

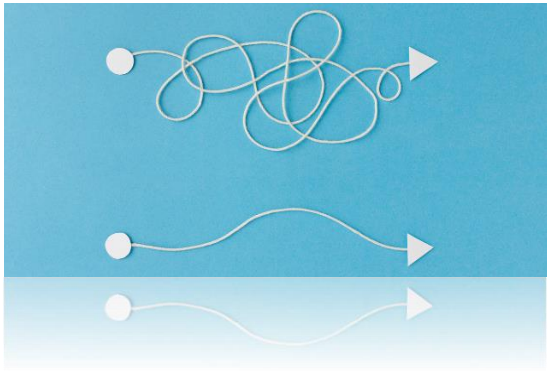


Image source: https://www.additudemag.org

Overfitting Problem

How to avoid Overfitting?

- *Method#4: Early Stopping*
  - A simple approach applicable to all **ANNs** (due to utilizing **GDA**)
  - But before going further, what is **Regularization?**

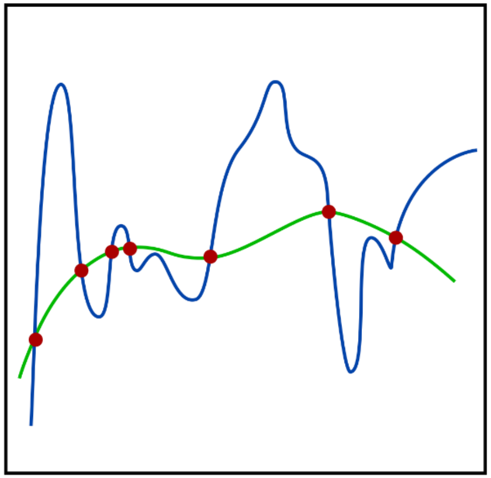


Image source: https://www.wikipedia.org

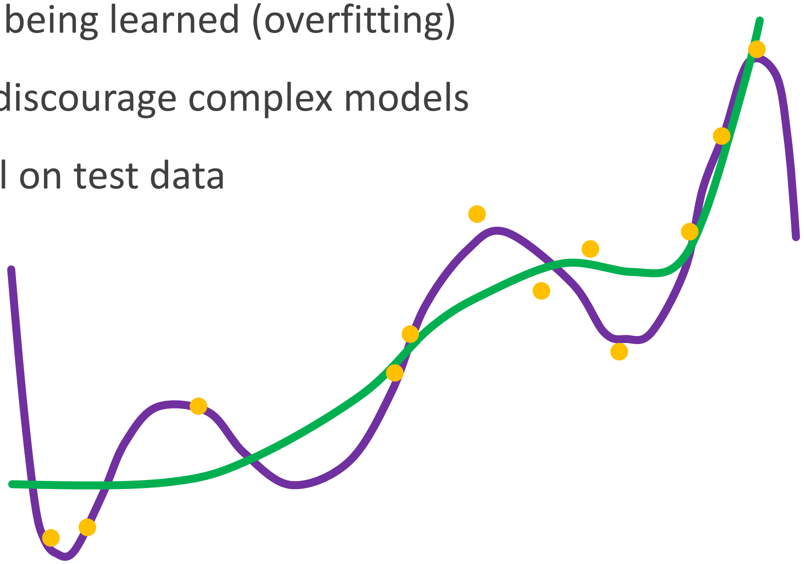
Overfitting Problem

Regularization

- Simply, trying to reduce the error by fitting a function on the training set
- Finding patterns in the training data and generalize them as much as possible

How can it help?

- Preventing complex information from being learned (overfitting)
- Limiting the optimization problem to discourage complex models
- Improving generalization of our model on test data



Overfitting Problem

How to avoid Overfitting?

- *Method#4: Early Stopping – continued*
  - A simple approach applicable to all **ANNs** (due to utilizing **GDA**)
  - Stop training when the **Generalization Error** increases!
    - *The error measured with respect to dev set often shows a decrease followed by an increase*
    - Stop when the training data starts to diverge from the testing data
  - Do the mentioned check in each iteration

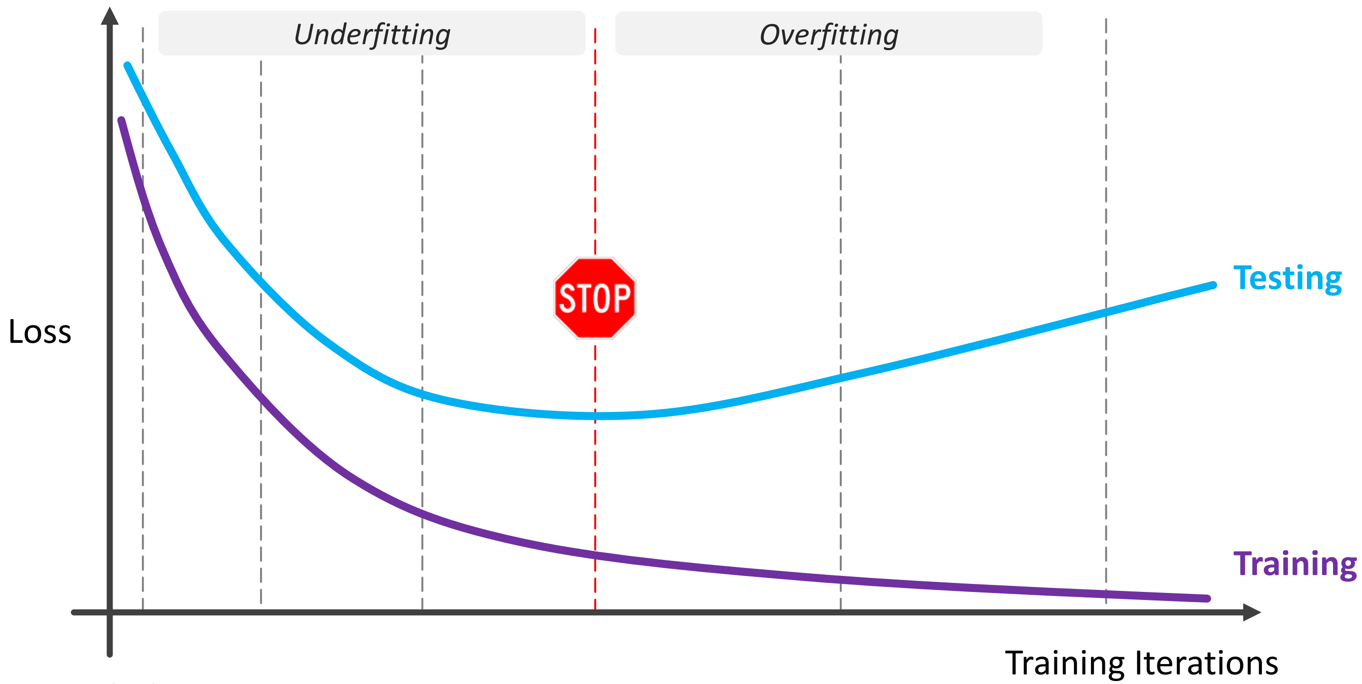
Monitoring the model's performance

Using a trigger to stop training

Overfitting Problem

How to avoid Overfitting?

- *Method#4: Early Stopping – continued*



Overfitting Problem

How to avoid Overfitting?

- *Method#4: Dropout*
  - A *per-layer regularization method* that randomly ignores some nodes
    - **Randomly setting AFs to zero** (params: **randomization probability**, e.g. 0.2)
  - Making the training process noisy to encourage the network to learn instead of memorize patterns
  - Reduces the capacity of the network (thinning)
  - Can be used on all/any of the hidden layers

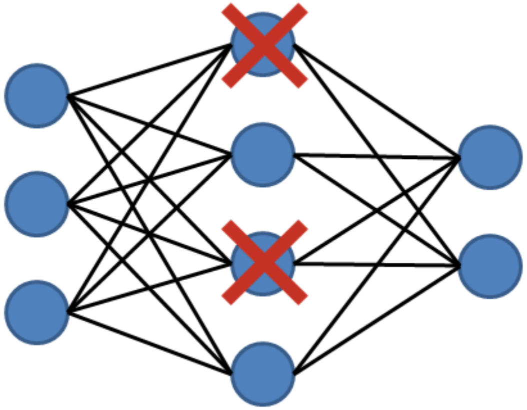
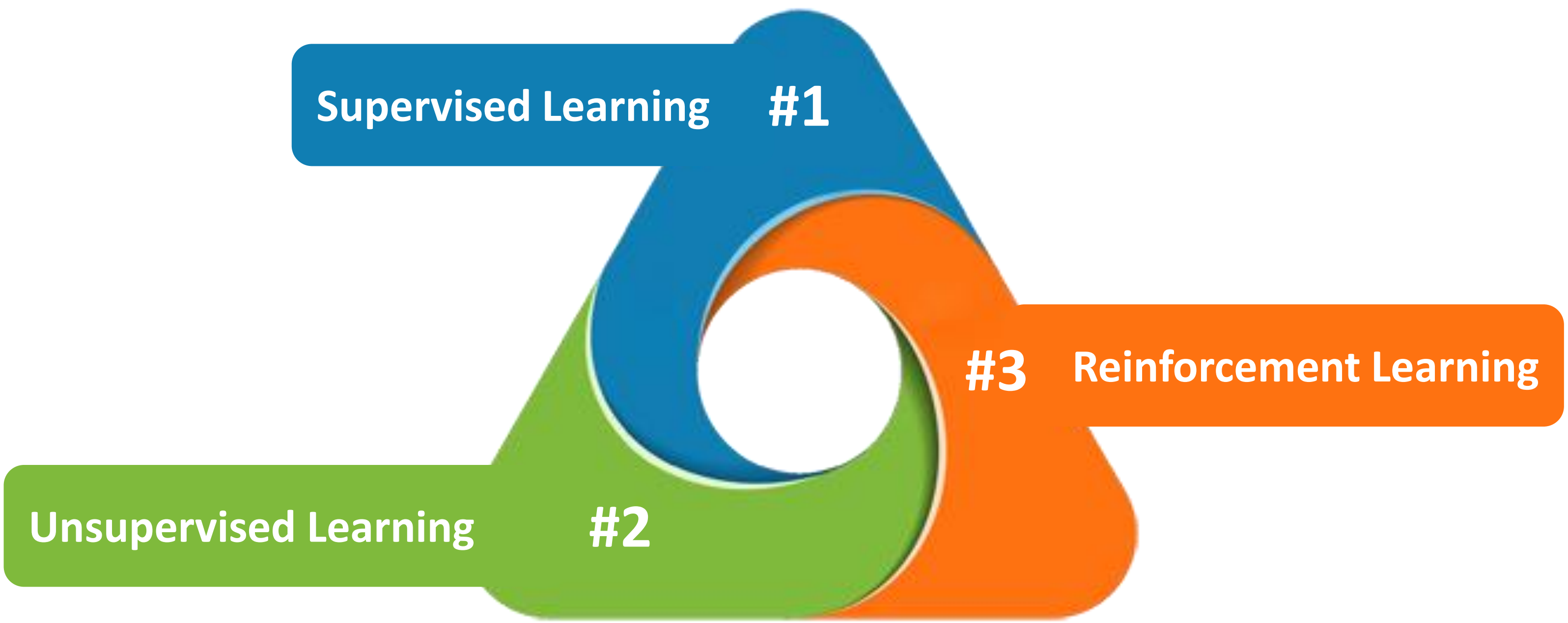


Image source: https://www.towardsdatascience.com

Learning Paradigms

Do different ANNs (and particularly, DNNs) learn in a unique manner?

- Definitely Not! It all depends on the learning task



Learning Paradigms

Supervised Learning

- Learning with a teacher! *(tagged data)*
- A set of paired inputs and outputs
  - **Goal:** producing desired output for each input sample
- Providing **feedback** on the quality of solutions
- Applications in classification, regression, pattern recognition, etc.

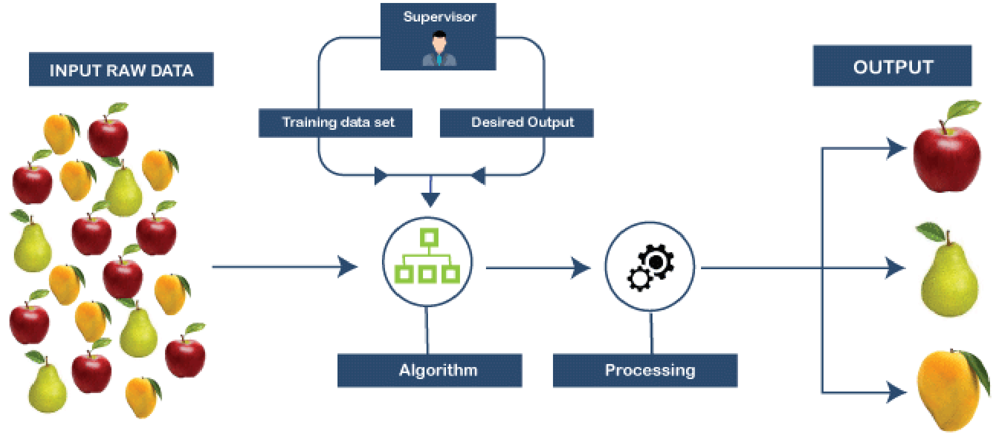


Image source: https://www.tutorialandexample.com

Learning Paradigms

Unsupervised Learning

- Input data + output data + error (cost) function
- Trying to learn patterns from untagged data
- The outputs are imaginative in most cases
  - Case study: organizing photos in a gallery (how?)
- **Note:** *DNNs are mainly impactful on structured data*

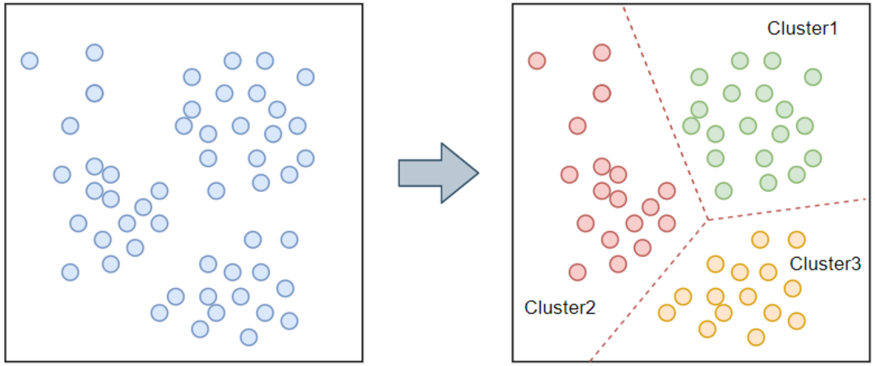


Image source: https://www.ecloudvalley.com

Learning Paradigms

Reinforcement Learning

- Trying to maximize the total (cumulative) reward
  - An agent learns to achieve a goal in an environment
  - Based on Reward and Penalty
- The model itself should find the solution with a maximized reward
  - Finding a solution with the lowest possible costs in future
  - Maybe even with trial and error
- **Advantage:** gaining experience from hundreds of evaluations

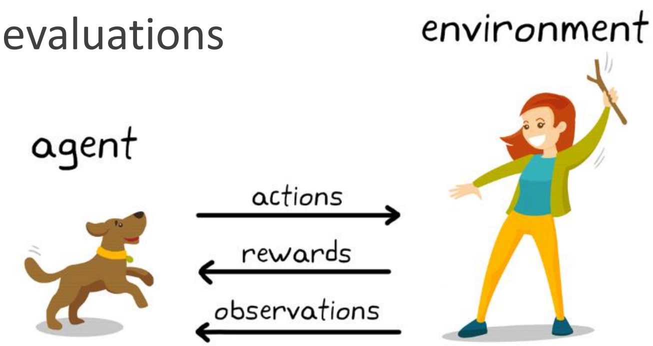


Image source: https://www.mathworks.com

Learning Paradigms

Reinforcement Learning

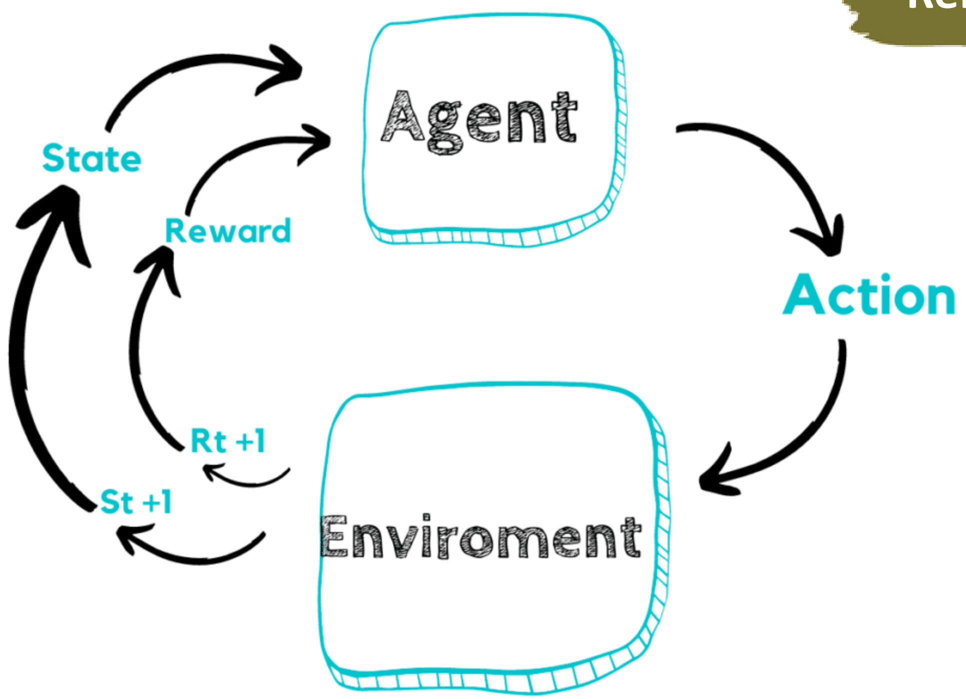


Image source: https://www.kitrum.com

Learning Paradigms

Reinforcement Learning

Case Study:
In autonomous vehicles, the AI module learns from the system's awards and penalties for unseen scenarios.

Image source: https://www.automate.org

Roadmap

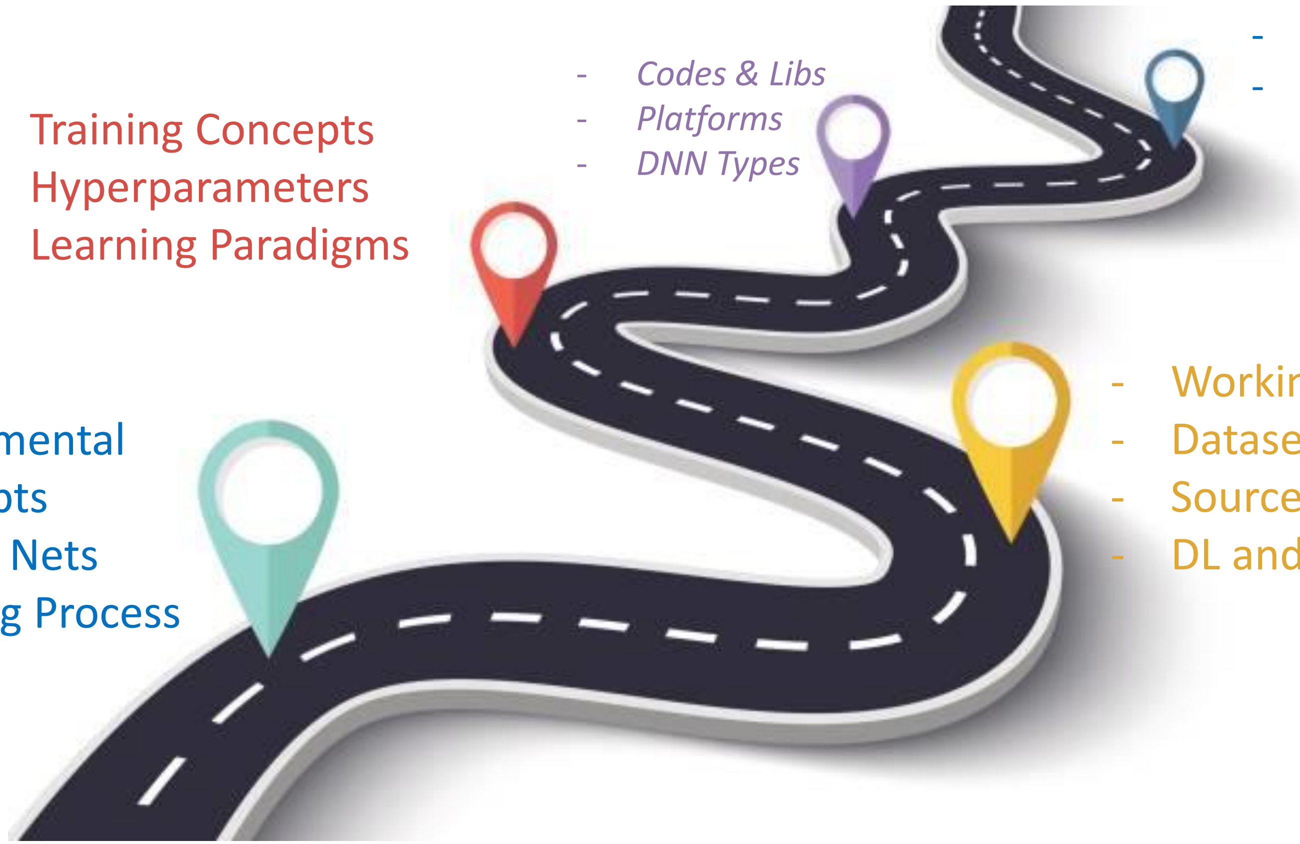
- Fundamental Concepts
- Neural Nets
- Training Process

- Training Concepts
- Hyperparameters
- Learning Paradigms

- *Codes & Libs*
- *Platforms*
- *DNN Types*

- *Projects?*
- *Papers?*

- Working with Data
- Datasets
- Sources of Data
- DL and Data



References

Web pages and Articles

- https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9
- https://towardsdatascience.com/what-are-hyperparameters-and-how-to-tune-the-hyperparameters-in-a-deep-neural-network-d0604917584a
- https://towardsdatascience.com/what-are-overfitting-and-underfitting-in-machine-learning-a96b30864690
- https://www.kdnuggets.com/2019/12/5-techniques-prevent-overfitting-neural-networks.html
- https://towardsdatascience.com/regularization-an-important-concept-in-machine-learning

References

Web pages and Articles

- https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide/
- https://en.wikipedia.org/wiki/Artificial_neural_network
- https://www.jeremyjordan.me/hyperparameter-tuning/

Papers

- N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov (2014) **Dropout: A Simple Way to Prevent Neural Networks from Overfitting**, Journal of Machine Learning Research, vol. 15, pp. 1929-1958.

Questions?