Hyper parameter Tuning

We'll start with learning rate:→

Learning Rate → It has highest impact on training
↓
Hidden Units
↓
Mini Batch Size (Higher Mini Batch, Lower Mini Batch)
↓
Beata → Beta value for optimizer
↓
Numbers of Layers
↓
Learning Rate Decay
↓
etc---

Random method is better than grid → (we pick some subset and we try to find some another subset).
because grid method is much more time consuming, it's a exhaustive search → (you have to find all possible soln)

Babysitting one model v/s Parallel Model Training

Panda or Caviar Approach (Caviar Approach better than Panda approach).

It will take one model and try to optimize that one model as max as possible. But if you make in initial model then your entire effort are waste.	Also known as parallel approach. It starts with multiple random model. Then you choose subset of that model which is performing better. Then you'll find the another combination in that model.

Baysien Appsoach is used for hyper parameter optimization

Keras Tuner

Swarm Optimization

(A group of agents or organism.

Swarm intelligence can be defined as the social behaviours of a swarm in which autonomous individuals interact with each other in a decentralised and self-organised manner.

PSO →

Particle Swarm Optimization

Population Based Stochastic Algo

inspired by social behaviour of bird flocking or fish schooling

Genetic operator (crossover mutation)

Random Population
fitness value
Update population

Algo of PSO is simple. Particles are numbes of simple entities in a search space. We create a population of particles and measure their individual fitness with an objective function of the problem. Particles are then moved from their current to the next position based on their personal best location, and on the swarm's best location so far. By iterating the moves the swarm gradually

NAS ~~X~~ mini batch size ~~berg~~ NO [PSO, LAMP, fedral AI, Adversial ⊛ Pseudocode ⊙, ~~hyper~~ Attack, hyperparameter]

reaches an optimal point of the objective function over generations.

⇒ LR Learning rate: controls the rate or speed at which model learns. Specifically it controls the amount ~~at~~ of appositioned error than the weights of the model are updated with each time they're updated.

Large LR → learn faster small LR → ~~m mod~~ may allow model to learn more optimal.

Hidden Units → The no. of the hidden units is the main measure of model's learning capacity

Large Hidden (↑) model overfitting

Mini Batch Size: → It has an effect on the resource requirements of the training process, speed and no. of iterations.

Large Batch size → allows computational boosts that utilizes matrix multiplication.

Small batch size → add more noise to data.

Beta: → also called momentum and ranges from 0→1. It sets the weights b/w the average of previous values and current value ~~of~~ to calc. the new weighted avg.

No. of Layers → Increasing no. of layers is helpful in some cases expect CNN.

	Large value	Small value
LR	Increase Regularization Incr training speed	Cause overfitting
Batchsize		
momentum		
Weight Decay		

Optimize Hyperparameters

↳ Grid Search

Try every possible combination of parameters. Expensive, time taking

Random Search → Random combinations of the values of hyperparameters are used to find the best soln. Can miss some imp. points in search space.

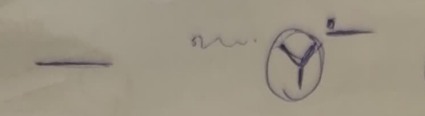
Bayesian model-based methods can find better hyperparameter in less time because they reason

In contrast to random or grid search, keep track of past evaluation results which they use to form a probabilistic model mapping hyperparameters.

Keras Tuner :→ Library to pick best/most optimal hyperparameters. Supports 4 types of tuners or algo.

→ Bayesian Optimization
→ Hyperband
→ Sklearn
→ Random Search.

Hyperband :→ It extension on top of successive halving algo
It solves problem of no of configuration and allocating the budget. It frequently perform successive halving frequently with diff budget to find best config.



28 × 2/9

Fedrated learning

↳ Decentralized form of machine learning. We'll train model on device themselves, and not on the centralized server, local data used to train our model.

→ centralized machine learning application will have a local copy, where users can use them acc. to our need.

Category	Description
→ Parameter pruning and quantization	reducing redudant parameters which aren't sensitive to the performance.
→ Low rank factorization	Using matrix/tensor decomposition to estimate the informative parameters
→	

Differential Privacy → It doesn't affect data subject. while maximizing utility/data accuracy for the queries.

→ DP guarantees that:→

(i) Raw data won't be viewed

(ii) Subjects privacy will be valued over mining insights from data.

Homorphic Encryption → Form of encryption that allows computation on encrypted data without decrypting it.

Adversarial Machine Learning is a technique which tries to modify an existing model, in order to introduce errors in predictions.

→ Black box Attack :→ the attacker has no knowledge of model.

→ Grey box :→ the attacker has partial knowledge of model, such as model architecture, parameters, training methods or training data.

→ White Box :→ attacker has complete knowledge of model.

Training time :→ Attack is to modify learning process. It has two categories :→

→ Data Accessing :→ Attacker have (partial) access to the dataset, thus they can create substitute model, which can be used in testing phase.

→ Poisoning :→ attacker modify dataset or model, in order to create altered trained model.

→ Evasion → Spoofing (Biometric)