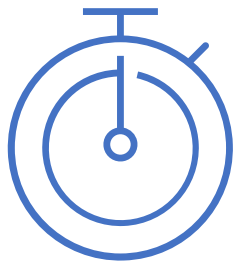


# Hyperparameter Tuning

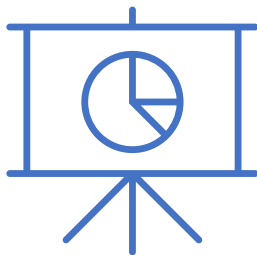
# Hyperparameter Tuning

## Introduction

Reasons for tuning parameters



training / inference time



improving results



convergence

# Hyperparameter Tuning

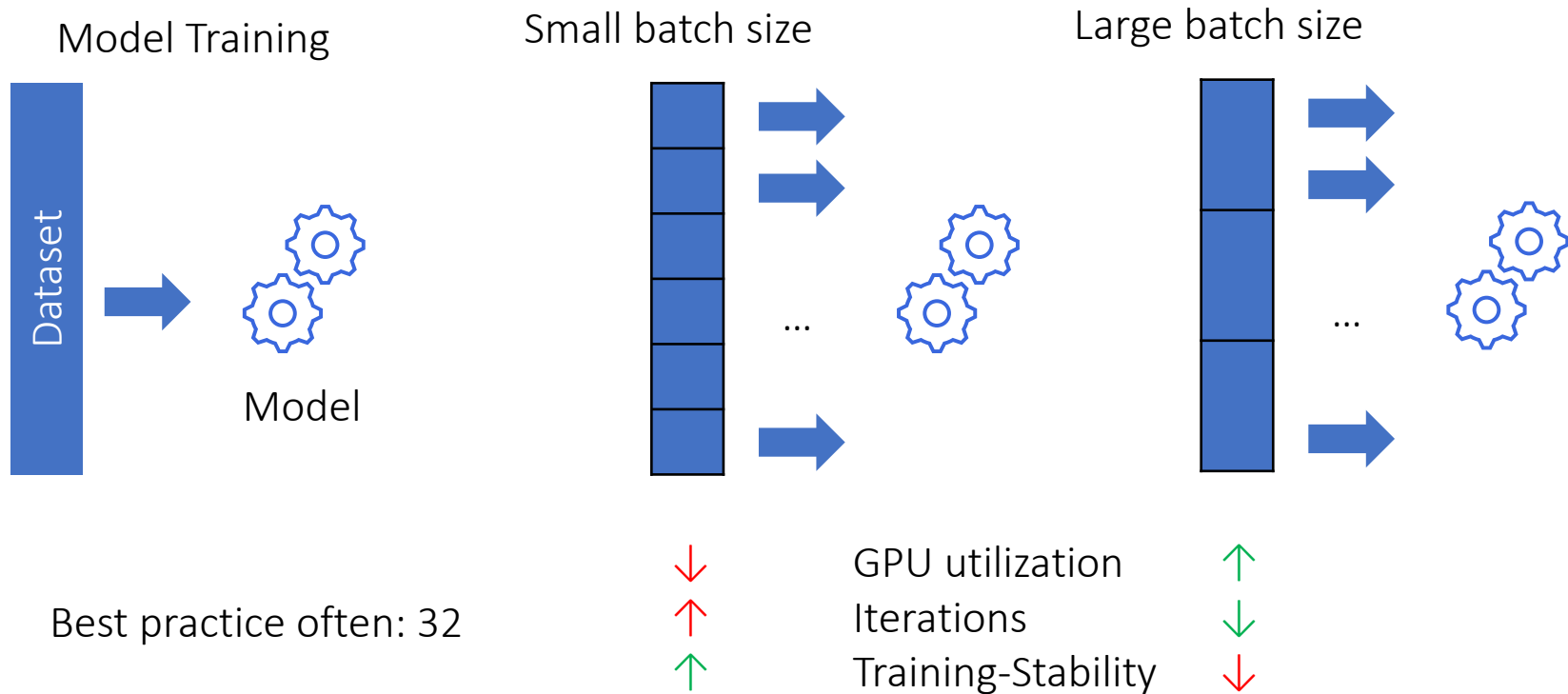
## Tunable Hyperparameters

- Hyper-parameters huge impact on model performance
- Intuition...check multiple combinations of parameters and pick the best
- available packages: RayTune, Optuna, skorch
- Hyperparameters

network topology	number of nodes	layer types	activation functions	...
network objects	loss function	optimizer		
model training	learning rate	batch size	number epochs	

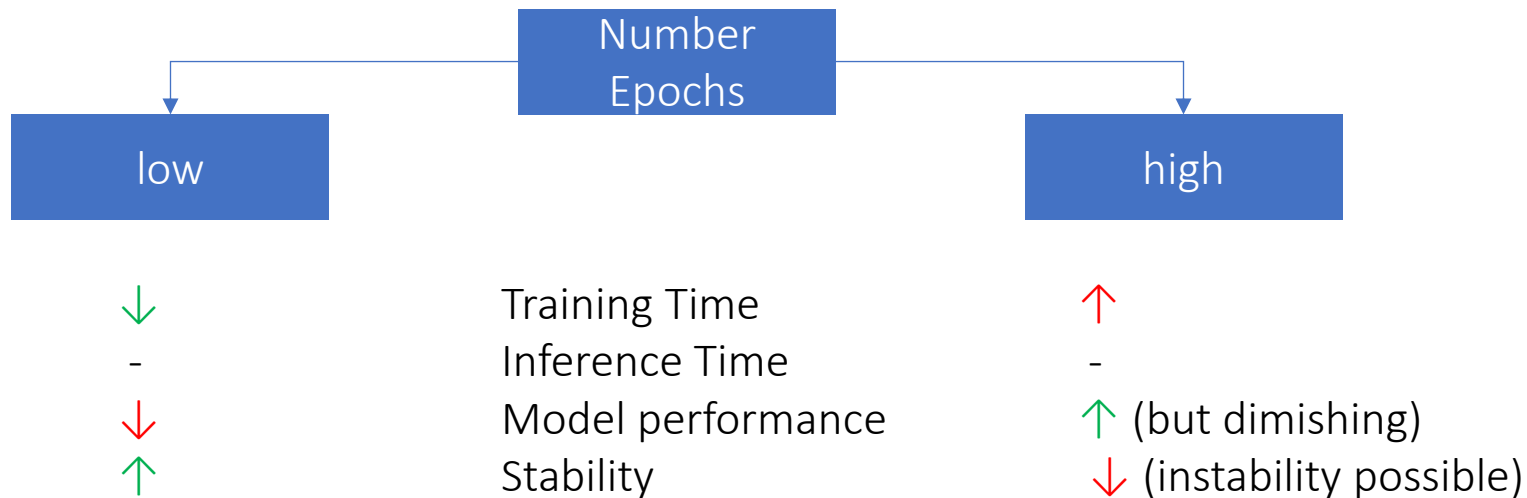
# Hyperparameter Tuning

batch size



# Hyperparameter Tuning

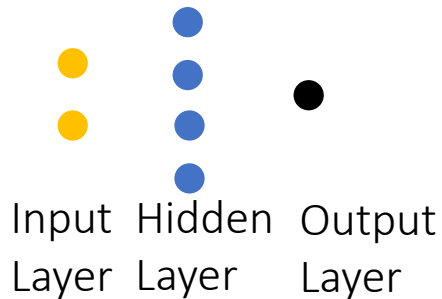
Epochs



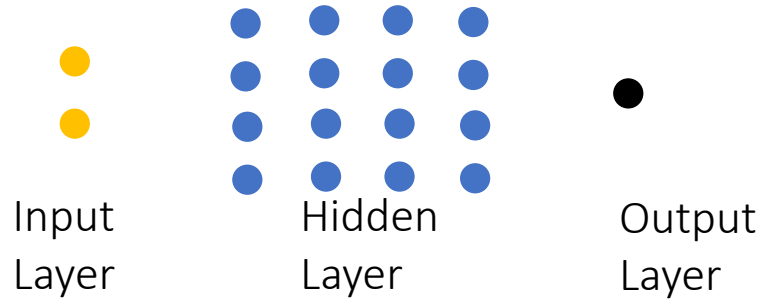
# Hyperparameter Tuning

## Hidden Layers

Few Hidden Layers



Many Hidden Layers



Ability to learn complex patterns



Training Time



Inference Time



Risk of Overfitting



# Hyperparameter Tuning

Nodes within a Layer

Few Nodes



Input Layer   Hidden Layer   Output Layer



Ability to learn complex patterns



Training Time

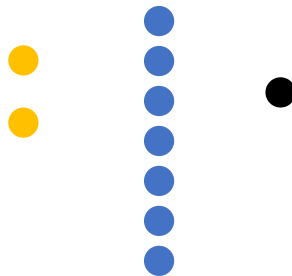


Inference Time



Risk of Overfitting

Many Hidden Layers



Input Layer   Hidden Layer   Output Layer



# Hyperparameter Tuning

## Types of Search

- grid search
  - define search space (set of parameters with limiting values)
  - evaluate each possible combination
  - e.g. `learning_rate = [0.1, 0.2]`,  
`batch_size = [2, 4, 8]`
  - good for checking well-known parameters
- random search
  - picks a point from configuration space
  - good for discovery

Run	learning_rate	batch_size
0	0.1	2
1	0.2	2
2	0.1	4
3	0.2	4
4	0.1	8
5	0.2	8



# Hyperparameter Tuning

skorch

- skorch...A scikit-learn compatible neural network library that wraps PyTorch.
- Repo: <https://github.com/skorch-dev/skorch>
- Works like a scikitlearn wrapper for PyTorch
- Can be integrated into
  - sklearn pipeline
  - grid search



```
from sklearn.model_selection import GridSearchCV

# deactivate skorch-internal train-valid split and verbose logging
net.set_params(train_split=False, verbose=0)
params = {
    'lr': [0.01, 0.02],
    'max_epochs': [10, 20],
    'module__num_units': [10, 20],
}
gs = GridSearchCV(net, params, refit=False, cv=3, scoring='accuracy', verbose=2)

gs.fit(X, y)
print("best score: {:.3f}, best params: {}".format(gs.best_score_, gs.best_params_))
```