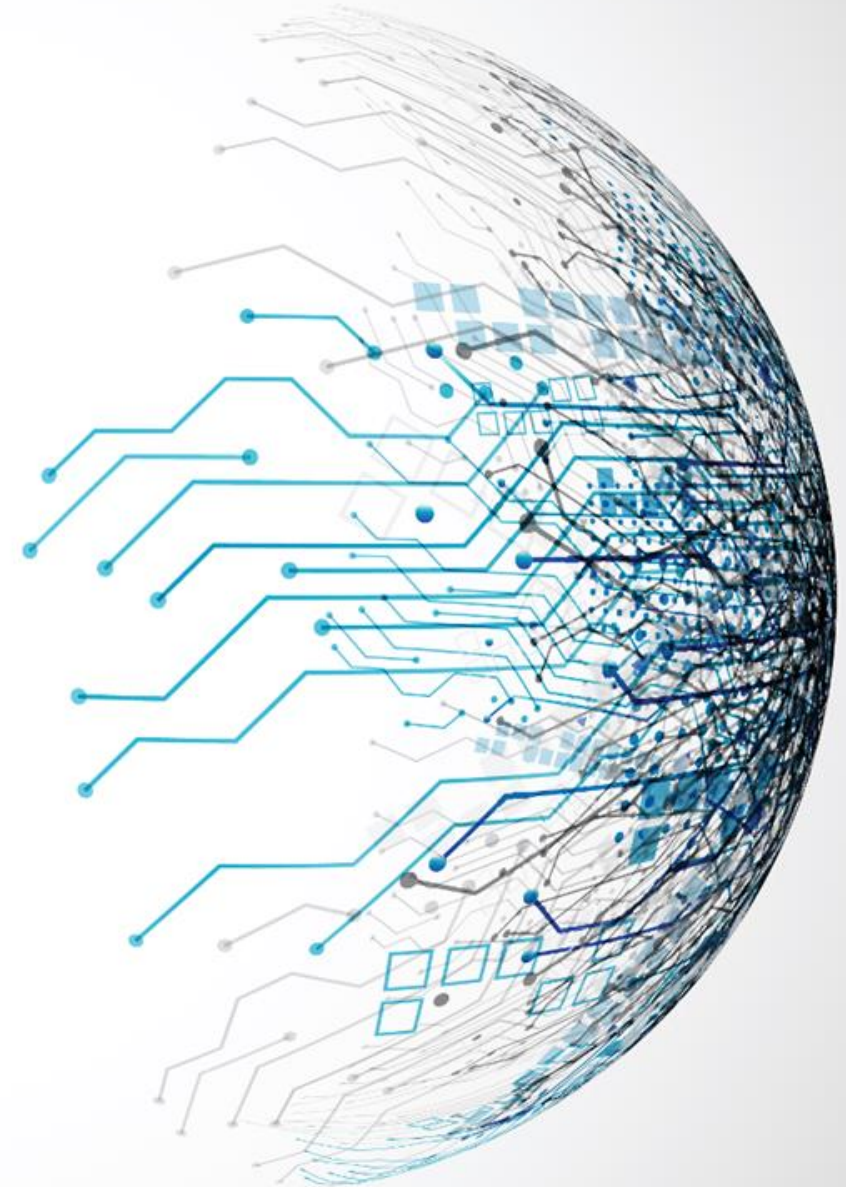


# Deep Learning

# Hyperparameter Optimization

Dr. Mohammed Salah Al-Radhi  
(slides by: Dr. Tamás Gábor Csapó)



# Copyright

Copyright © **Bálint Gyires-Tóth & Mohammed Salah Al-Radhi**, All Rights Reserved.

This presentation and its contents are protected by copyright law. The intellectual property contained herein, including but not limited to text, images, graphics, and design elements, are the exclusive property of the copyright holder identified above. Any unauthorized use, reproduction, distribution, or modification of this presentation or its contents is strictly prohibited without prior written consent from the copyright holder.

**No Recordings or Reproductions:** Attendees, viewers, and recipients of this presentation are expressly prohibited from making any audio, video, or photographic recordings, as well as screen captures, screenshots, or any form of reproduction, of this presentation, its content, or any related materials, whether during its live presentation or subsequent access. Violation of this prohibition may result in legal action.

For permissions, inquiries, or licensing requests, please contact: **{toth.b,malradhi}@tmit.bme.hu**

Unauthorized use, distribution, or reproduction of this presentation may result in civil and criminal penalties. Thank you for respecting the intellectual property rights of the copyright holder.

# References

<https://bit.ly/3AFIKuT>



# Announcements

## Project work

- Group and topic selection done

## Milestone 1: data acquisition, data preparation (+ optional: containerization)

- Deadline: 7th week, **Oct 15**, Tuesday, 23:59, moodle, GitHub repo
- Oct 16, Wednesday class: consultation about projects (required for each group)

## Milestone 2: baseline evaluation, baseline model

- Deadline: 11th week, **Nov 12**, Tuesday, 23:59, moodle, GitHub repo
- Nov 13, Wednesday class: consultation about projects (required for each group)

## Final submission

- Deadline: end of 14th week, **Dec 6**, Friday, 23:59, moodle, GitHub repo and documentation

# Outline

- Deep learning basics
- Incremental model development
- Initial model and manual fine-tuning
- Automatic hyperparameter optimization
- When is hyperopt not necessary?

A network diagram consisting of numerous nodes (small circles) connected by thin lines. The nodes are arranged in a somewhat horizontal line, with some branching out above and below. The lines are a light teal color. A white rectangular box is superimposed over the center of the image, containing the text "Deep learning basics".

# Deep learning basics





# Basic elements

- **Architecture:**
  - FC: Fully Connected
  - RNN: Recurrent Neural Network
  - CNN: Convolutional Neural Network
- **Activation functions:** to become intelligent
- **Forward pass:** predict
- **Loss:** how good is my model?
- **Backward pass:** identify where the model can be improved
- **Optimizer:** improve the model
- **Regularization:** work on unseen data

# References

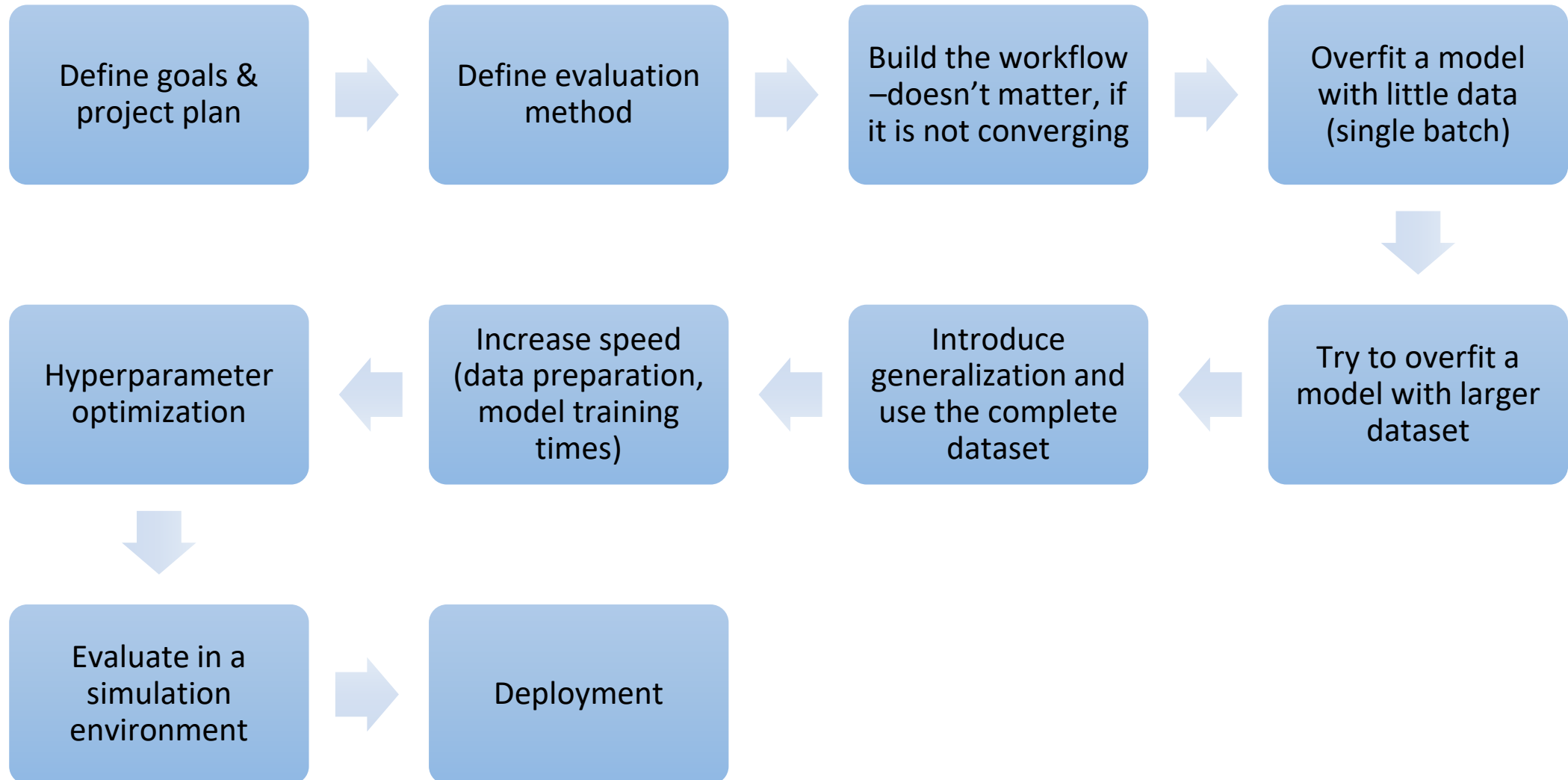
- Fully connected networks and activation functions:  
<https://cs231n.github.io/neural-networks-1/>
- Recurrent Neural Networks:  
<https://colah.github.io/posts/2015-08-Understanding-LSTMs/>
- Convolutional neural networks:  
<https://cs231n.github.io/convolutional-networks/>
- Backpropagation (forward and backward pass):  
<https://www.youtube.com/watch?v=bxe2T-V8XR&list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU>
- Loss functions and regularization:  
<https://cs231n.github.io/neural-networks-2/>
- Optimizers, early stopping, mini-batching:  
<https://www.deeplearningbook.org/contents/optimization.html>



A network diagram consisting of numerous nodes (small circles) connected by thin lines (edges). The nodes are arranged in a somewhat horizontal line, with some branching out above and below. The lines are a light teal color. A white rectangular box is superimposed over the center of the image, containing the text "Incremental model development".

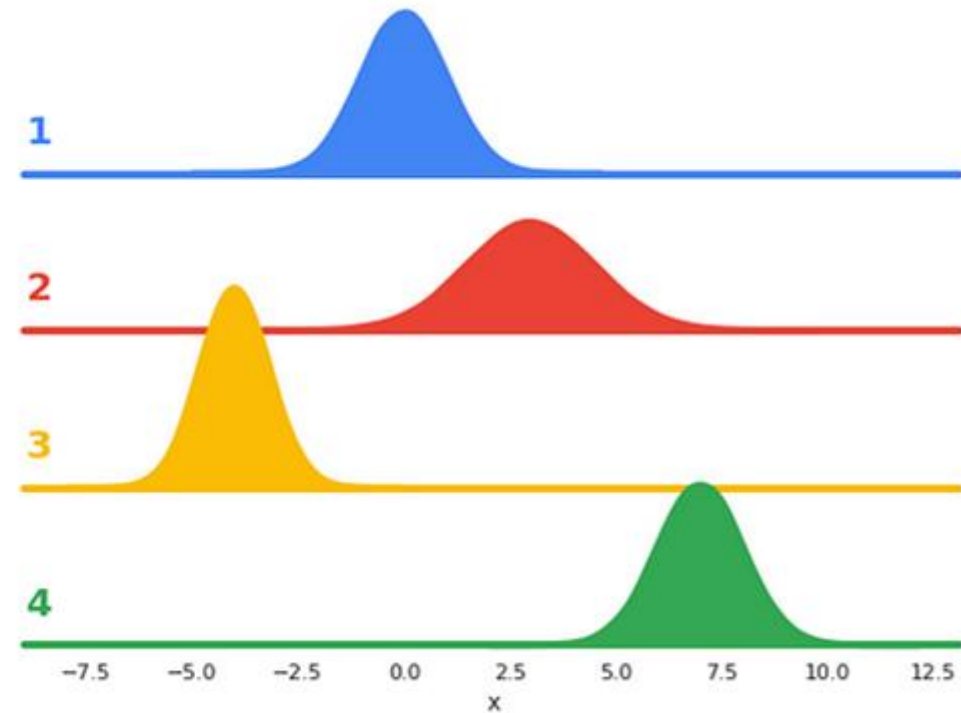
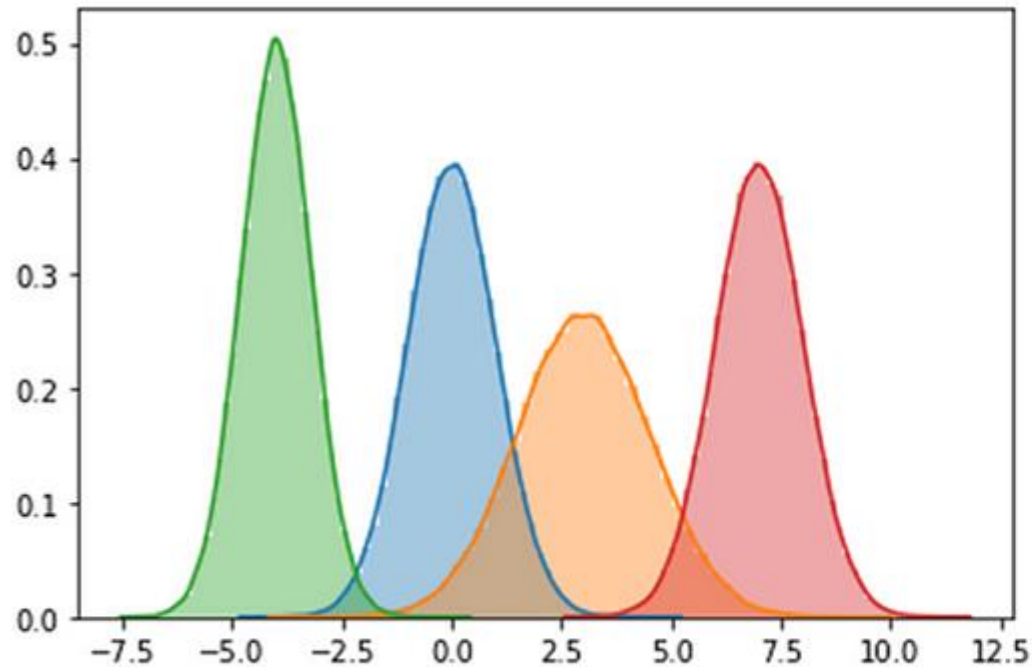
# Incremental model development

# Machine learning project main steps



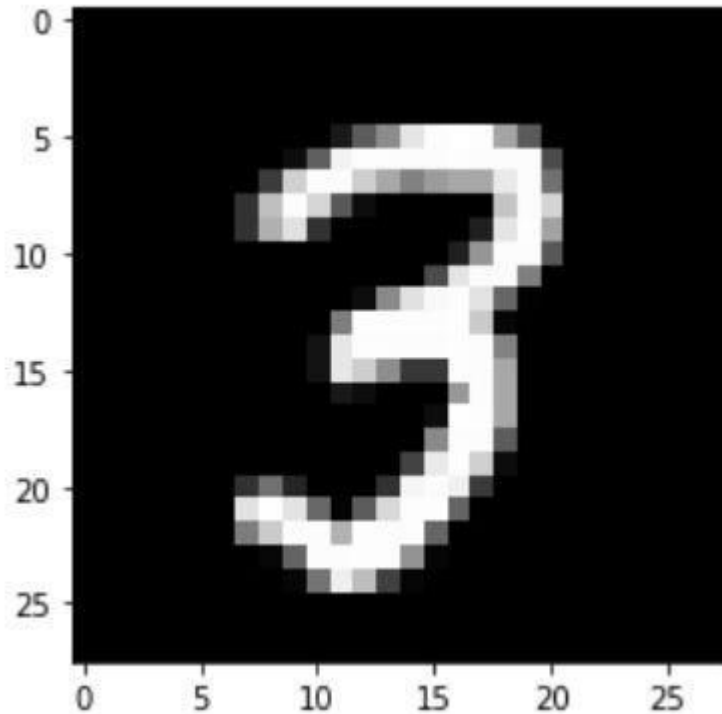
# Build the workflow

- Try to understand your data!
- Use a basic statistical model
- `sns.displot`



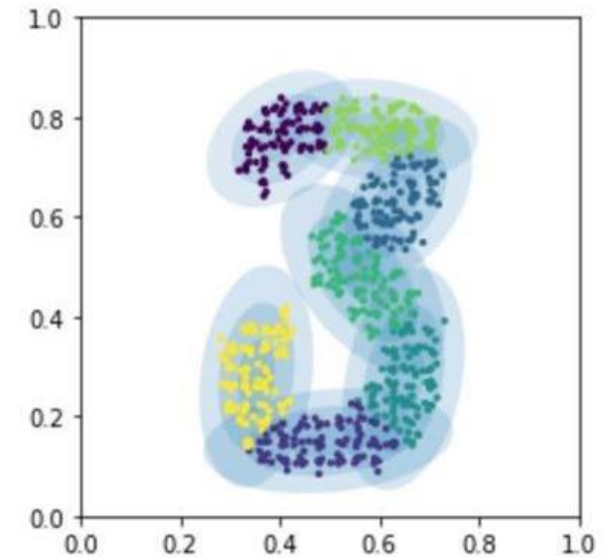
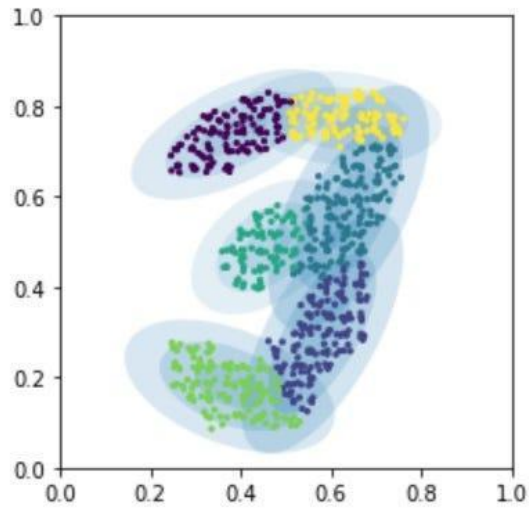
# Start with a simple neural network

- What can you observe on the „MNIST” image with your eye?



# Start with a simple neural network

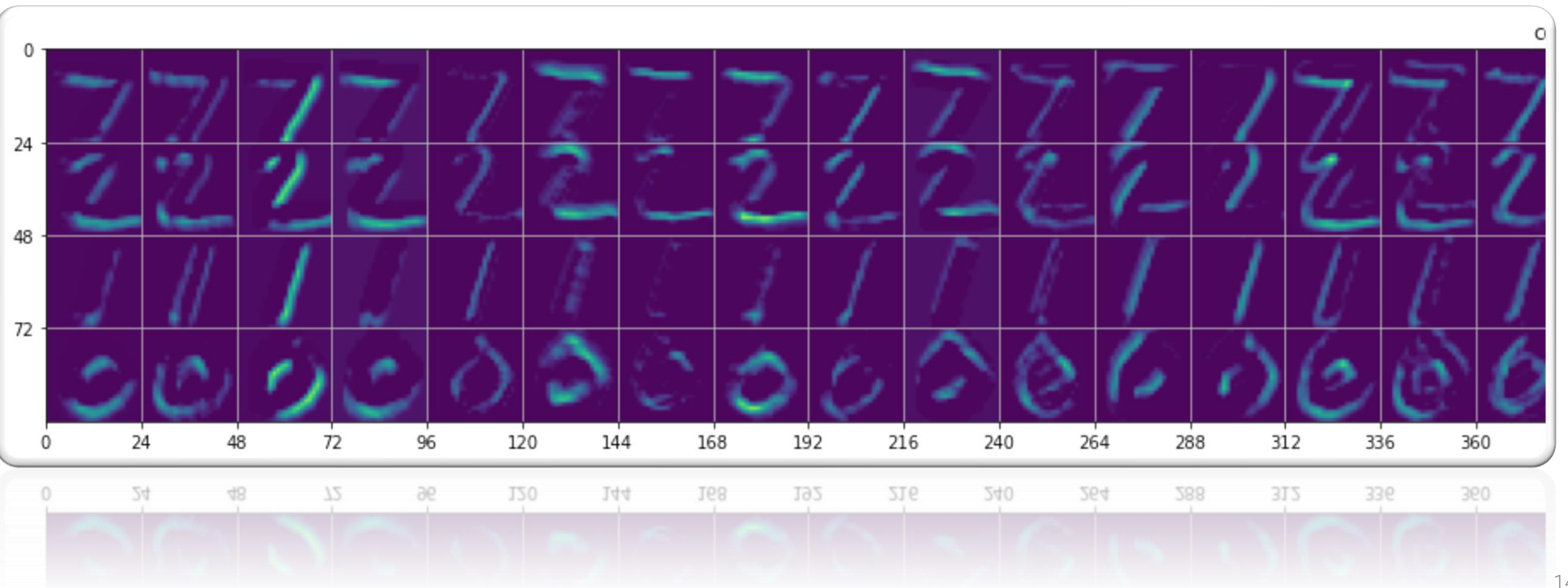
- What can a simple statistical model see?





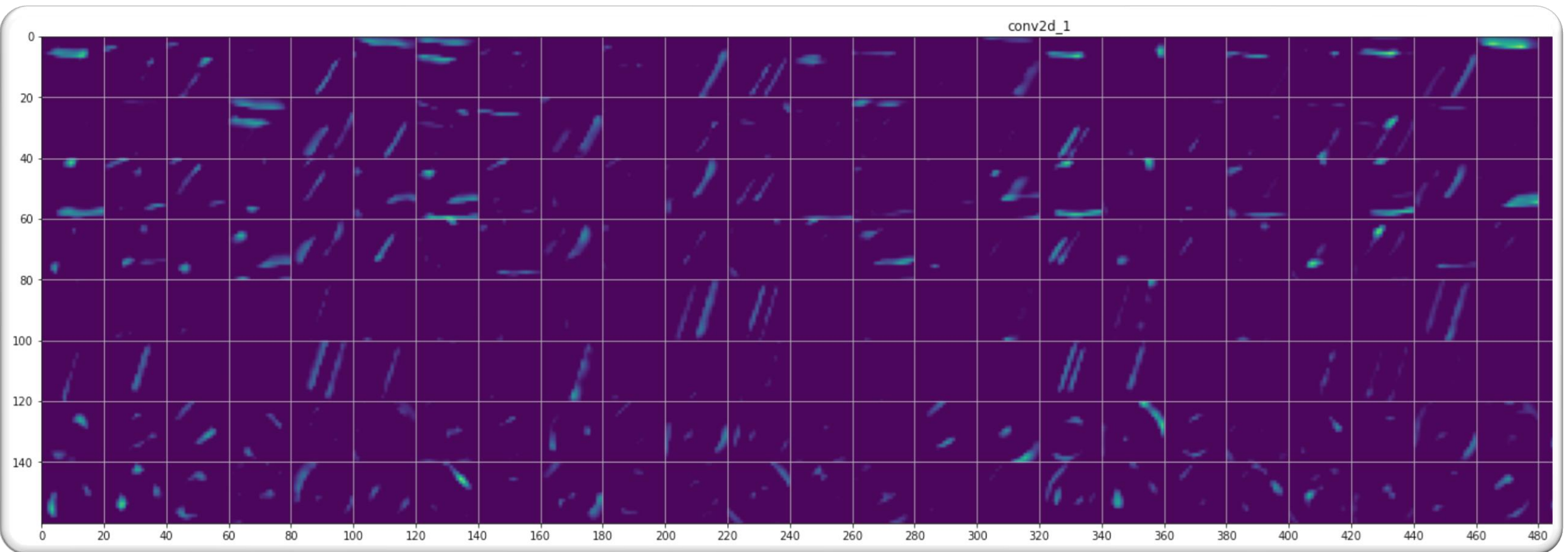
# Start with a simple neural network

- What can a simple CNN see? Check activations!



# Start with a simple neural network

- What can a simple CNN see? Check activations!



# Simple model .... more complex

- Start from a simple model, single batch of data
- Add more data
- Try more complex models

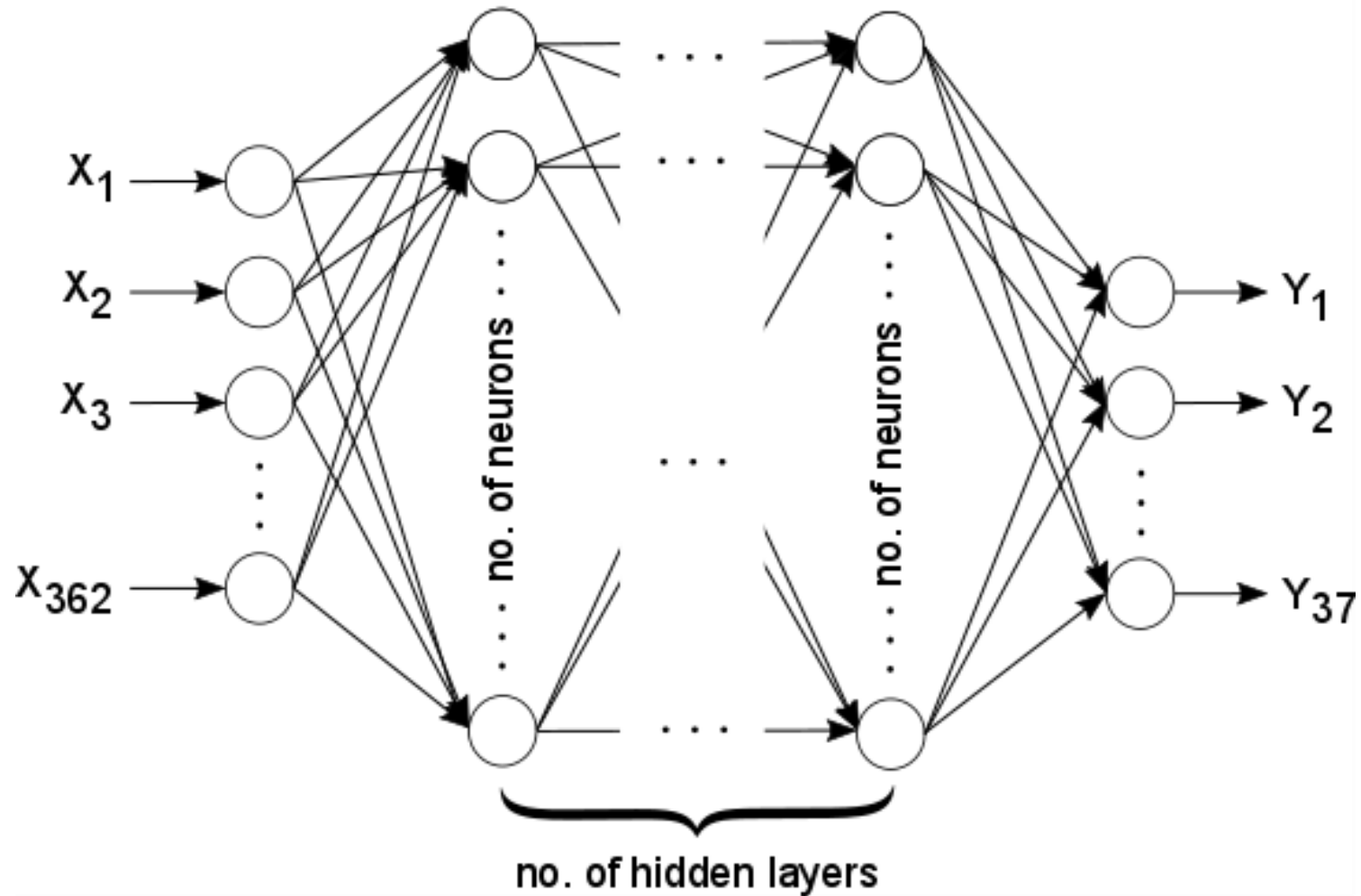
## Why automatic hyperopt?

- When manual hyperopt isn't feasible any more
- Tweak model performance
- Analyze several 1000 different parameter combinations

A background network diagram consisting of numerous nodes (small circles) connected by thin lines, forming a complex web. The nodes are colored in shades of teal and blue, and the lines are thin and light-colored. The network is spread across the width of the slide, with a higher density of nodes and connections in the center.

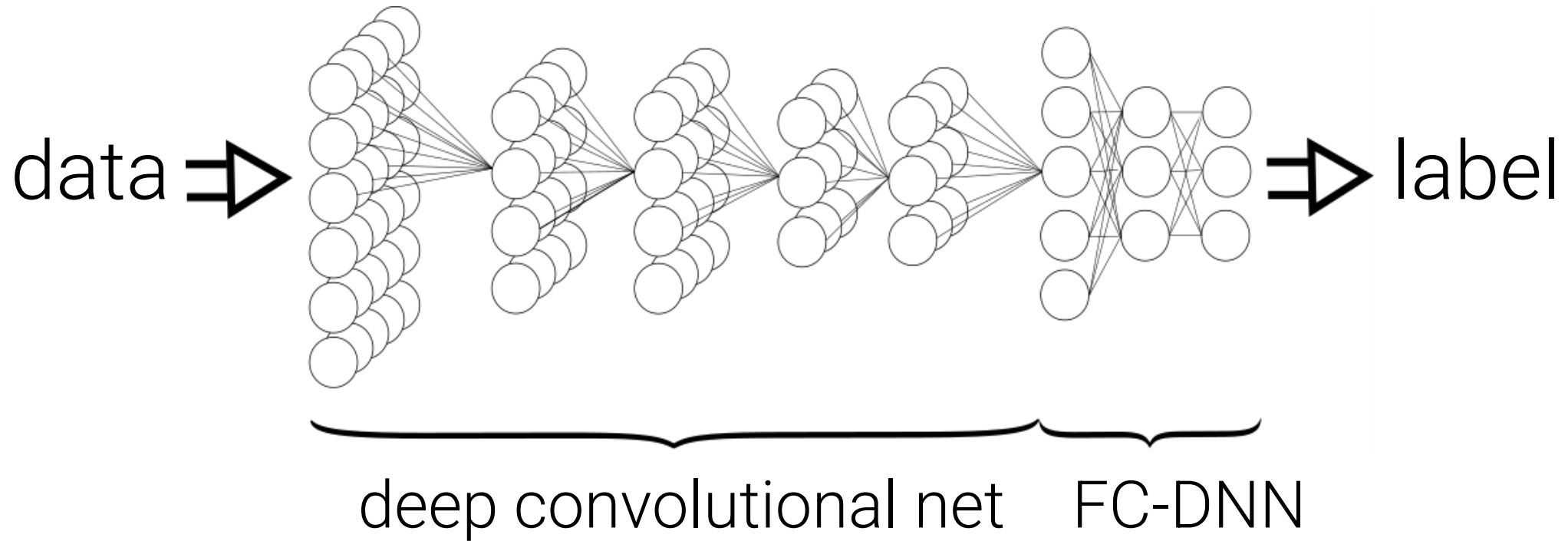
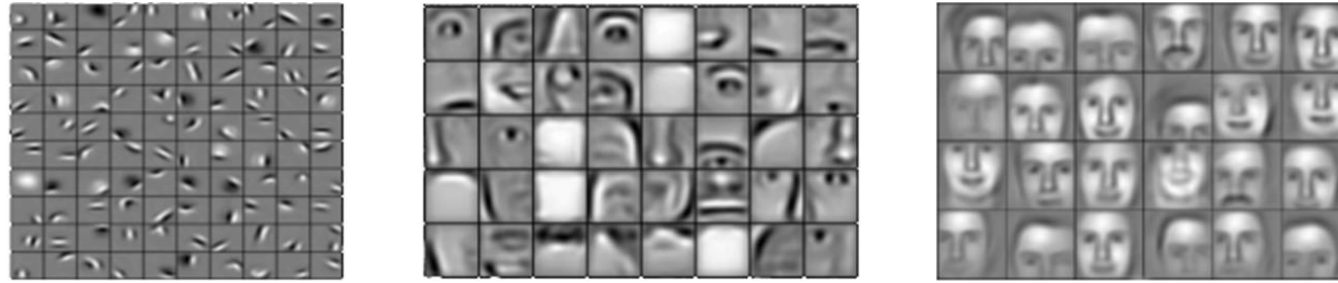
# What to hyperopt / 1: Network architectures

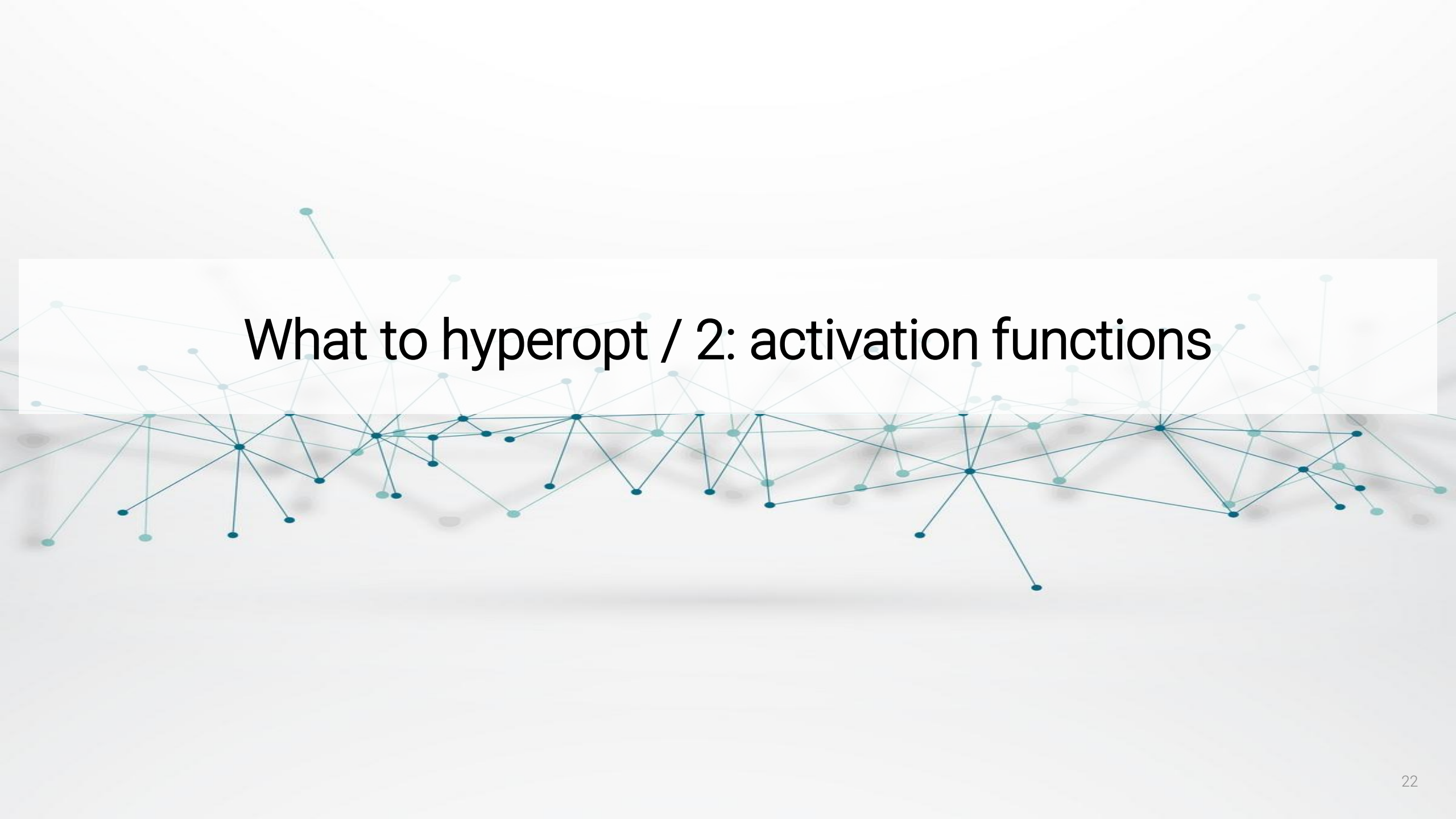
# FC = Fully Connected, Feed-forward





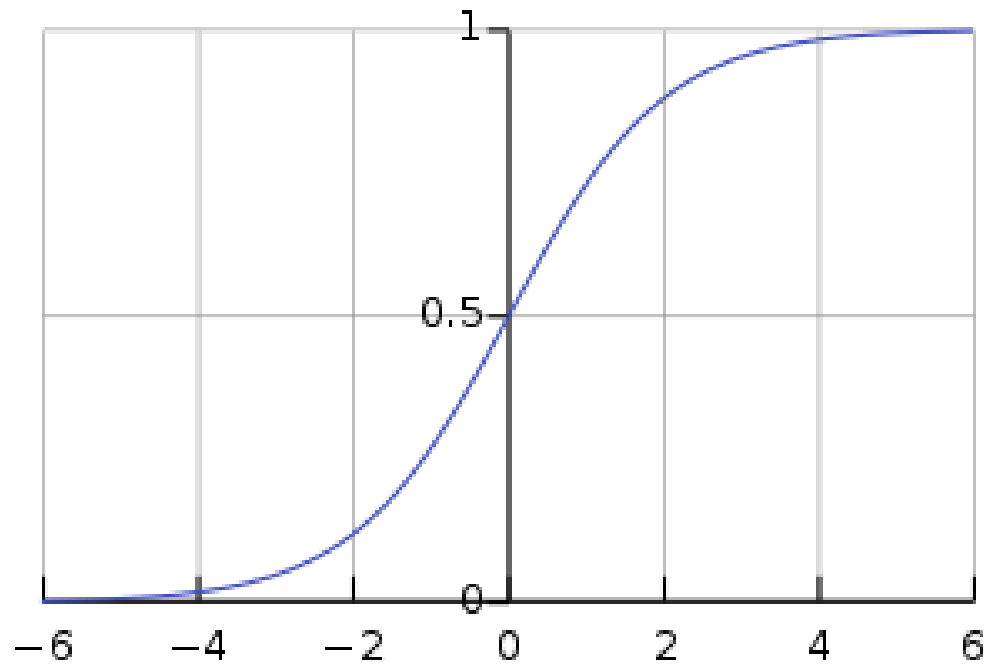
# Convolutional: 2D-CNN





# What to hyperopt / 2: activation functions

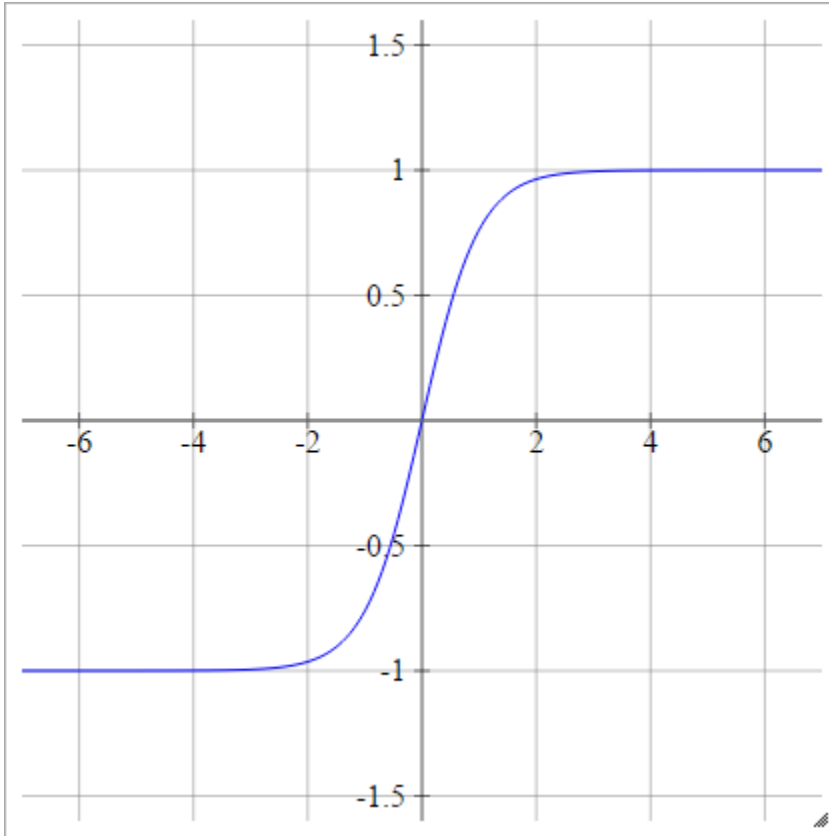
# Sigmoid



$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$

Source: [https://en.wikipedia.org/wiki/Sigmoid\\_function](https://en.wikipedia.org/wiki/Sigmoid_function)

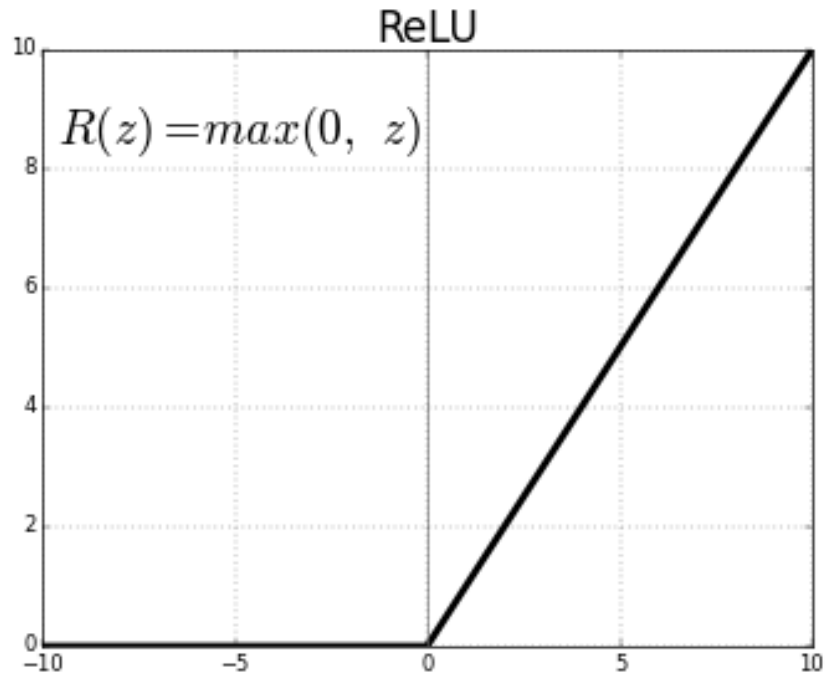
# Tanh



$$f(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$$

Source : <https://stats.stackexchange.com/questions/115258/comprehensive-list-of-activation-functions-in-neural-networks-with-pros-cons>

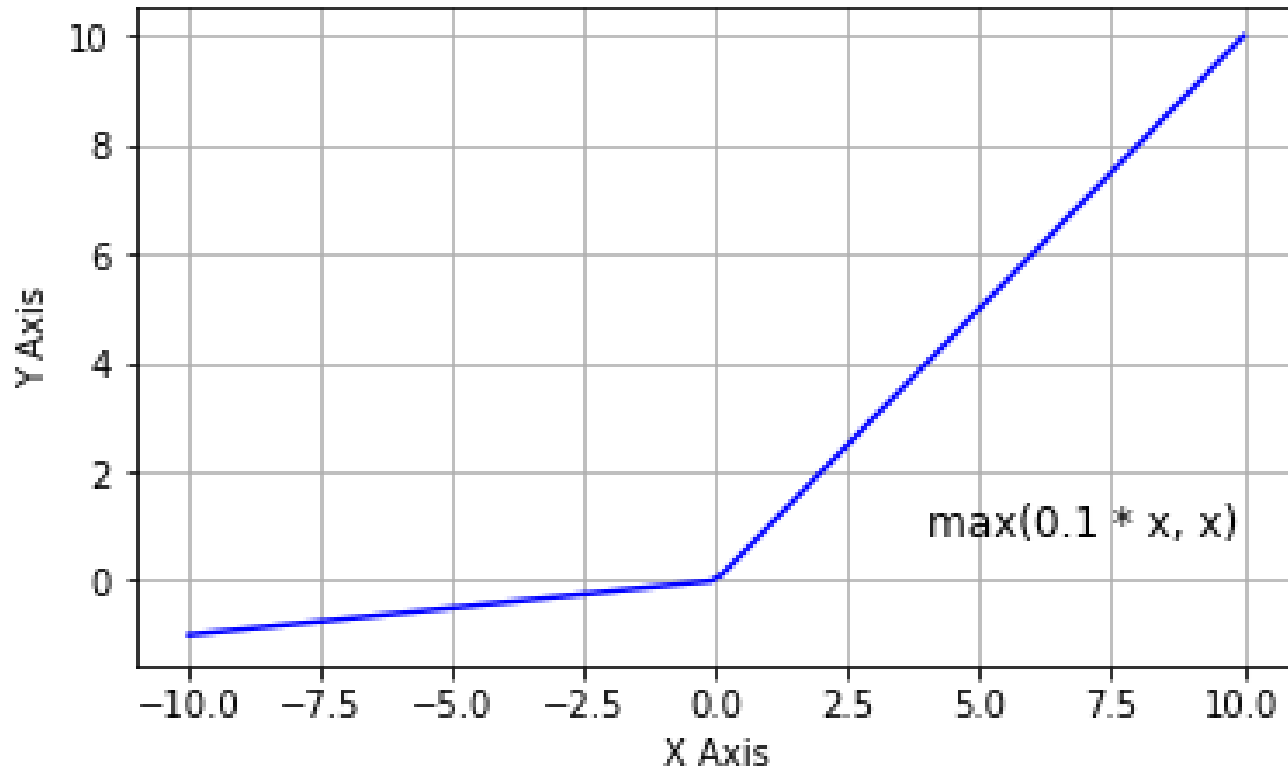
# ReLU



$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$



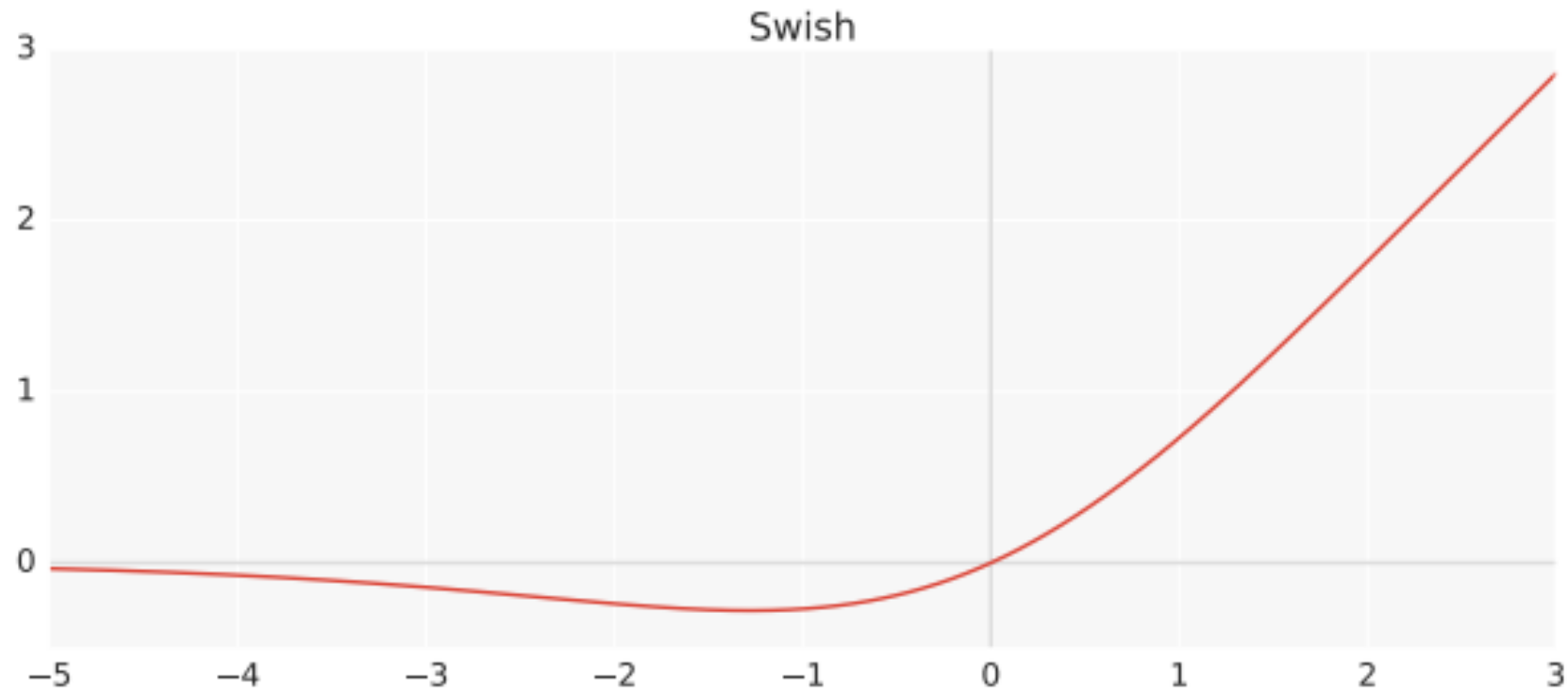
# Leaky ReLU, PReLU



$$f(x) = \begin{cases} 0.01x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$$

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ ax & \text{otherwise} \end{cases}$$

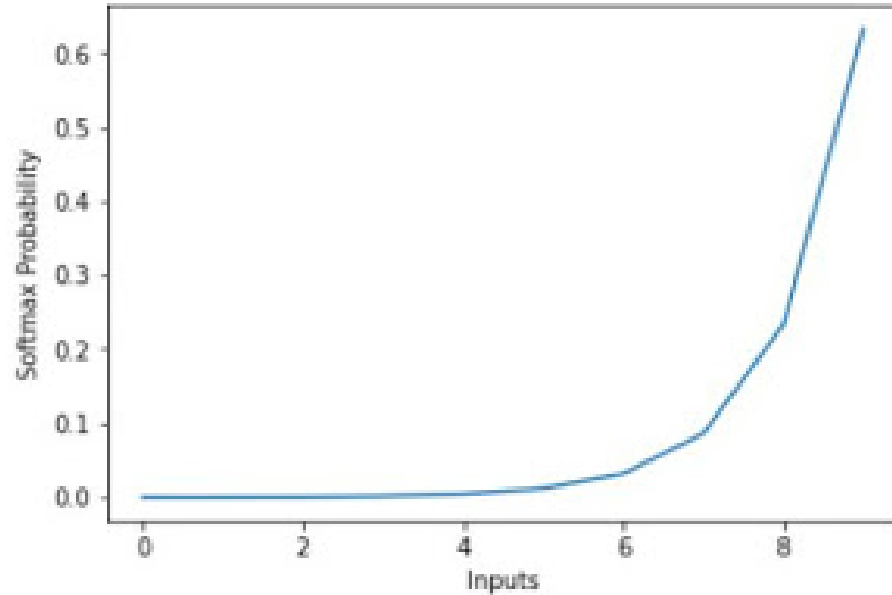
# SWISH



$$\begin{aligned} f(x) &= x * \text{sigmoid}(x) \\ &= x * (1 + e^{-x})^{-1} \end{aligned}$$


Source : <https://medium.com/@neuralnets/swish-activation-function-by-google-53e1ea86f820>

# Softmax



$z$  is a vector of the inputs to the output layer (if you have 10 output units, then there are 10 elements in  $z$ )

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}}$$

The background of the slide features a complex network diagram. It consists of numerous small, light blue circular nodes connected by thin, light blue lines. The nodes are distributed across the slide, with a higher density in the lower half. A semi-transparent white rectangular box is centered horizontally and vertically, containing the main title text. The overall aesthetic is clean and technical, typical of a presentation on machine learning or optimization.

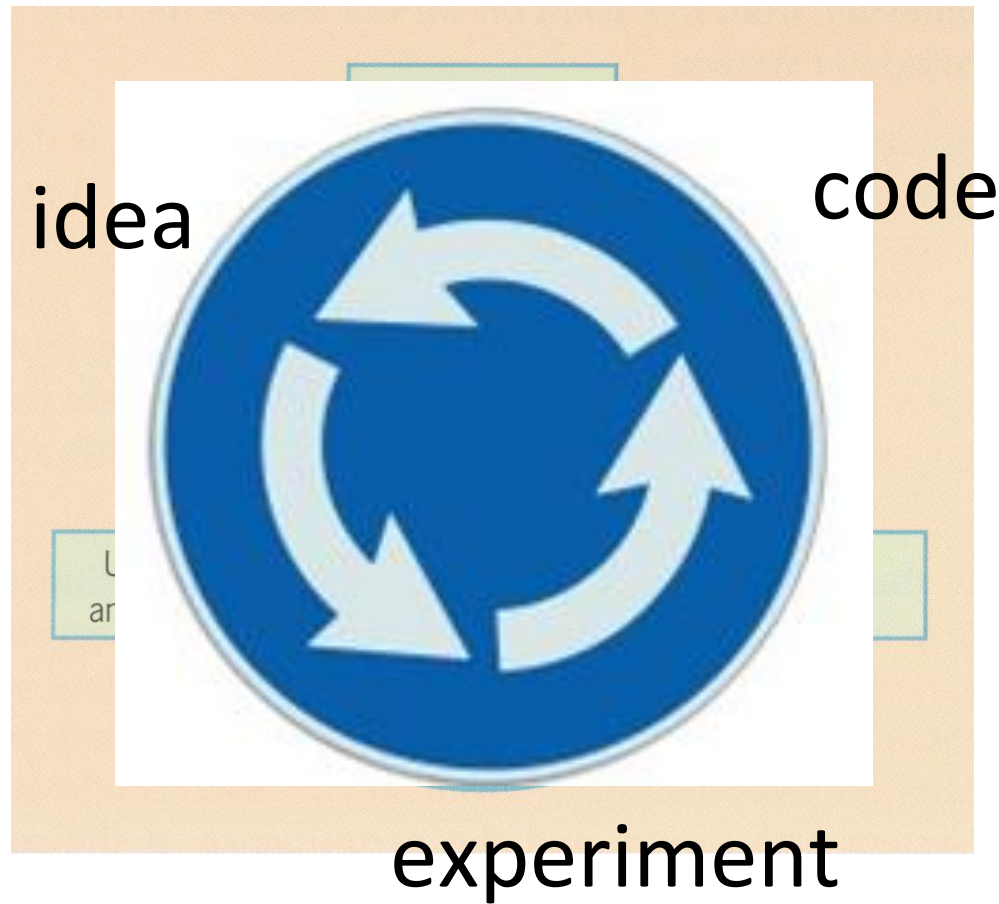
# What to hyperopt / 3: optimizer algorithms

# Optim

- torch.optim.
  - SGD
  - RMSprop
  - Adam
  - RAdam
  - NAdam
  - AdamW
  - Adadelta
  - Adagrad
  - Adamax



# ML/ DL: iterative process...



# Hyperparameters (for a simple FC-DNN)

- Epoch number
- Batch size
- Alpha Learning rate
- Beta Momentum
- Optimizer
- Number of layers
- Number of neurons
- Activation functions

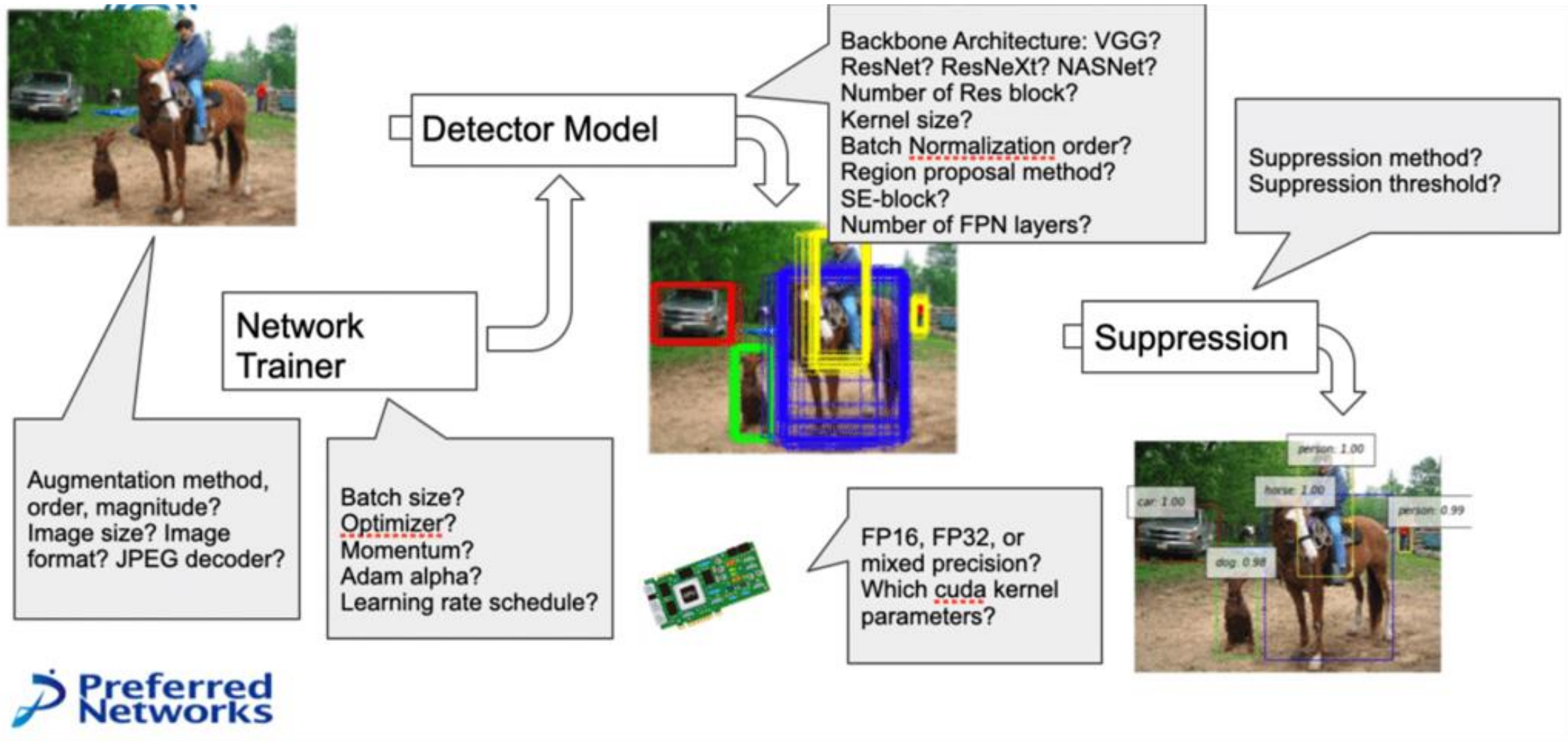
# Hyperparameters, linear scale

- Layers: 2, 3, 4
- No. of neurons: 100, 200, 300, ... 1000

# Hyperparameters, what scale?

- Alpha: 0.0001, 0.001, ... 1
- $10^{-4}$ ,  $10^{-3}$ , ...  $10^0$
- 10-base logarithm: back to linear scale, -4, -3, , ... 0

# ... and even more hyperparameters!



The image features a network diagram with numerous nodes and edges. The nodes are represented by small circles, and the edges are thin lines connecting them. The nodes are arranged in a somewhat horizontal line, with some branching out above and below. The edges are thin and light blue. A white rectangular box is overlaid on the center of the image, containing the word "Generalization" in a bold, black, sans-serif font. The background is a light, textured gray.

# Generalization



# Generalization



Training set (labels known)



Test set (labels unknown)

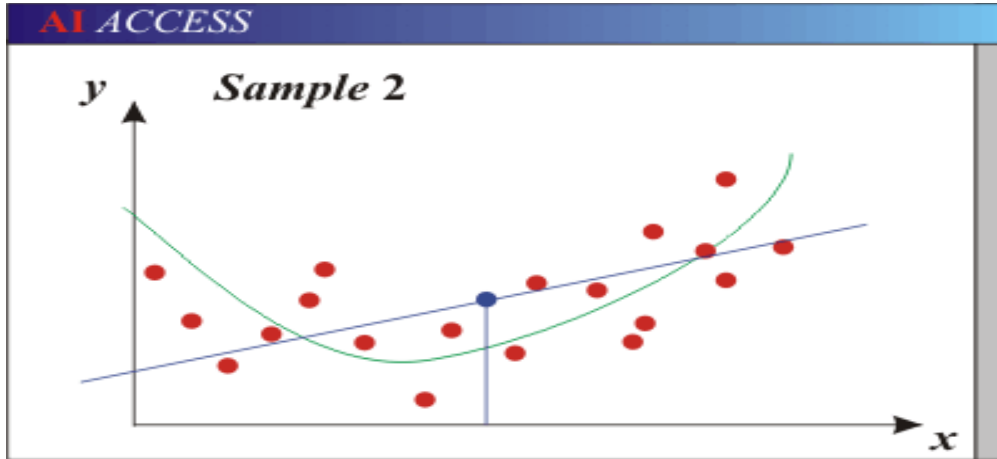
- How well does a learned model generalize from the data it was trained on to a new test set?

# Generalization

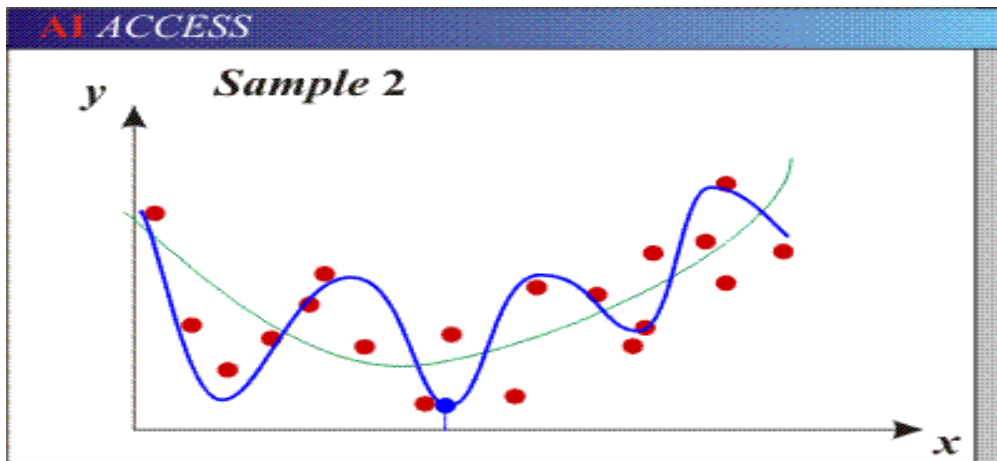
- Components of generalization error
  - **Bias:** how much the average model over all training sets differ from the true model?
    - Error due to inaccurate assumptions/simplifications made by the model
  - **Variance:** how much models estimated from different training sets differ from each other
- **Underfitting:** model is too “simple” to represent all the relevant class characteristics
  - High bias and low variance
  - High training error and high test error
- **Overfitting:** model is too “complex” and fits irrelevant characteristics (noise) in the data
  - Low bias and high variance
  - Low training error and high test error



# Bias-Variance Trade-off

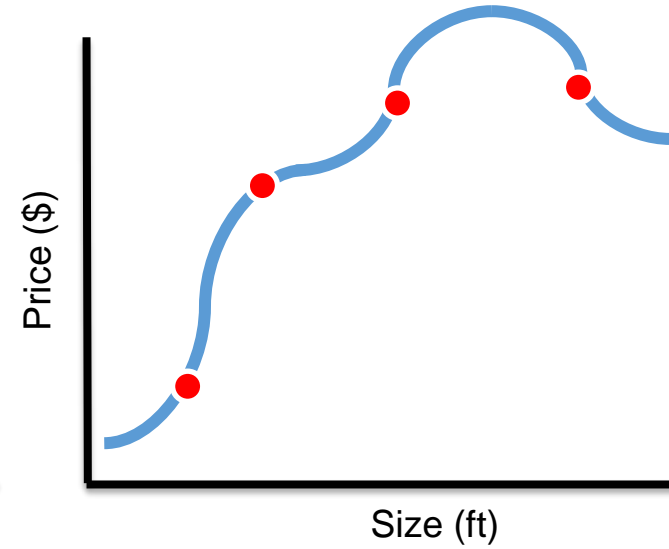
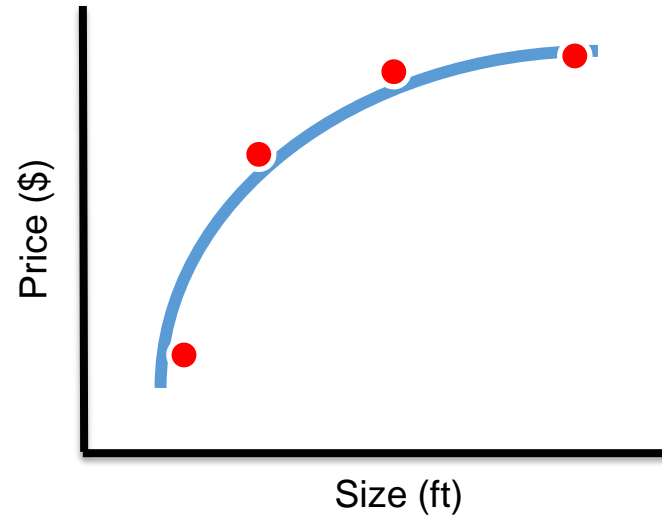
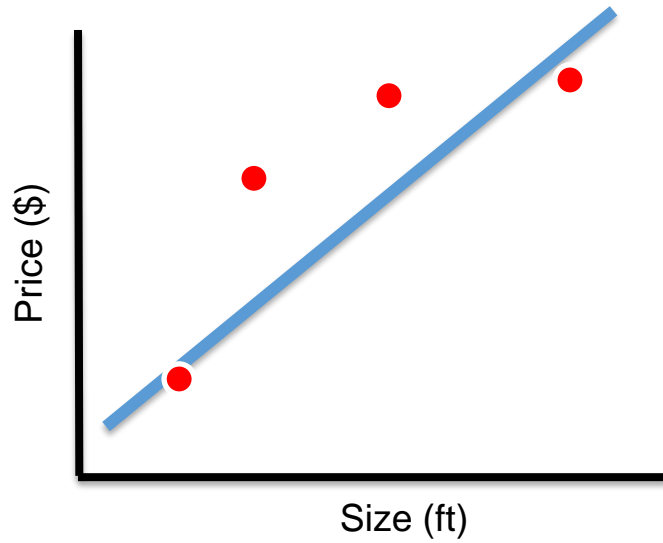


- Models with too few parameters are inaccurate because of a large bias (not enough flexibility).

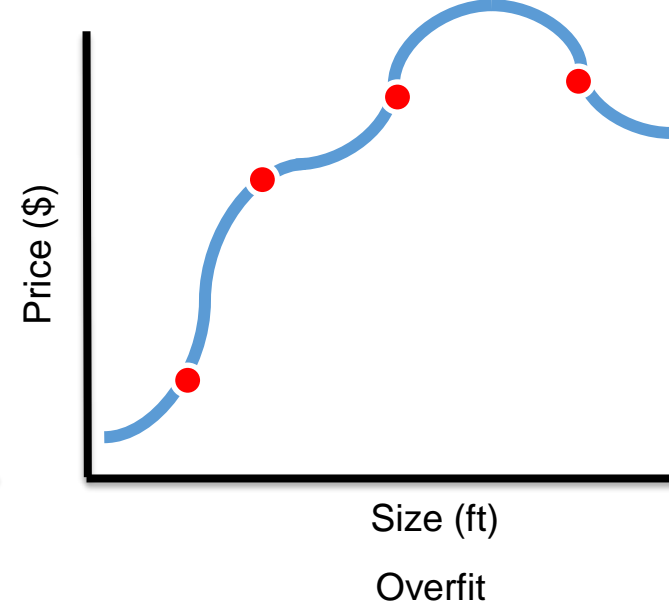
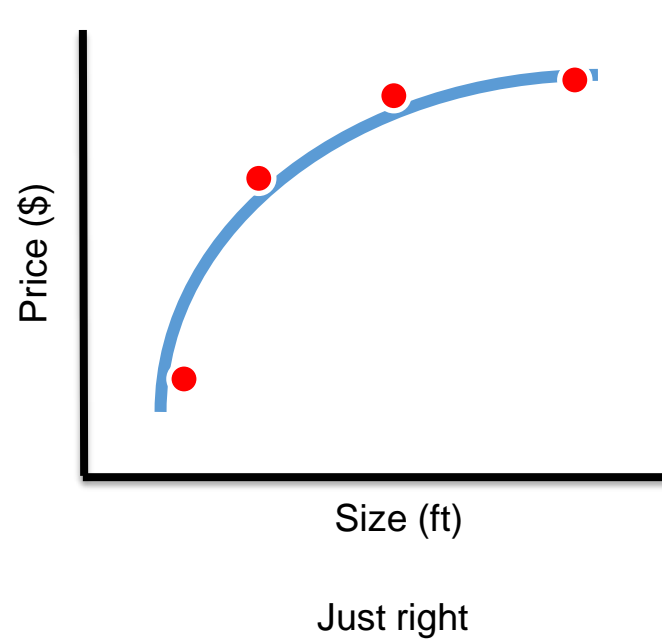
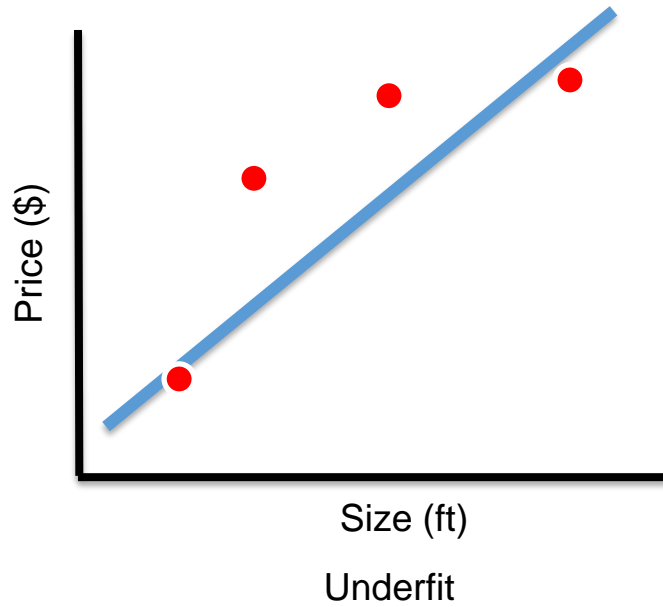


- Models with too many parameters are inaccurate because of a large variance (too much sensitivity to the sample).

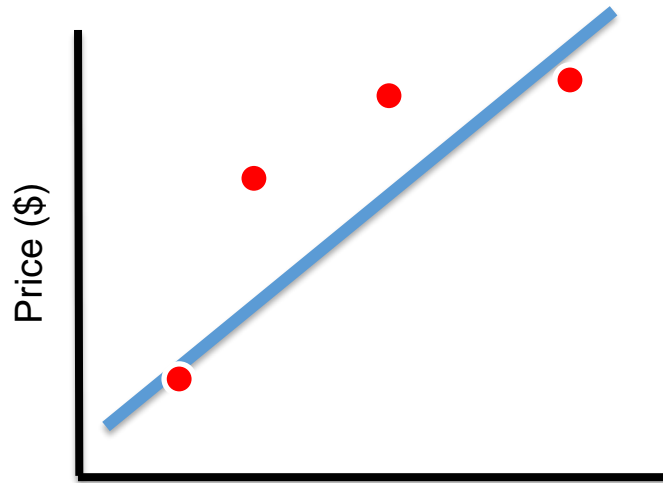
# Bias/Variance Trade-off



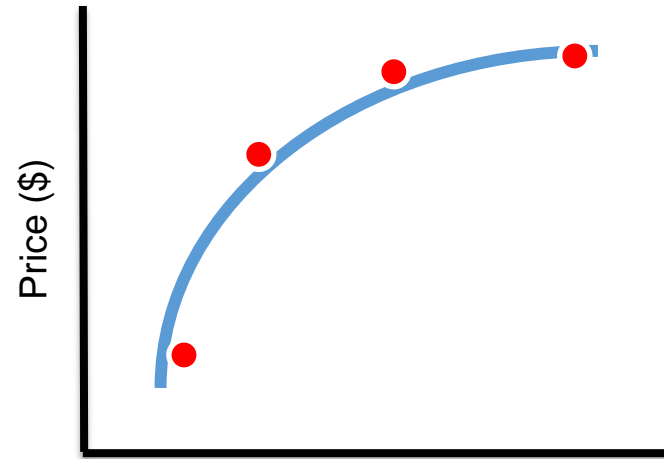
# Bias/Variance Trade-off



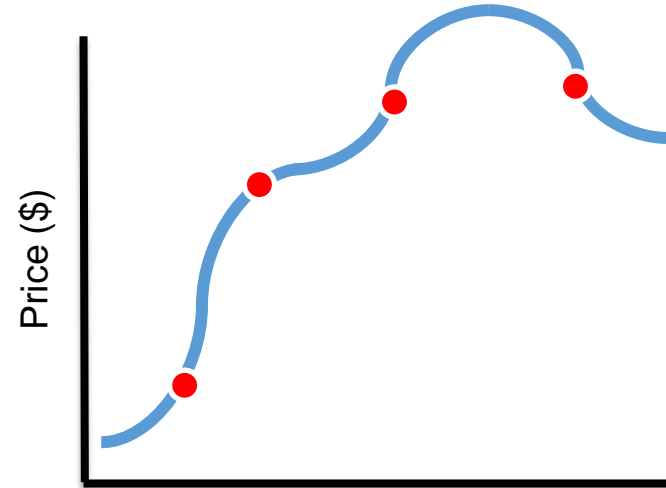
# Linear Regression with Regularization



Size (ft)  
Underfit  
High bias  
Too simple  
Too much regularization



Size (ft)  
Just right



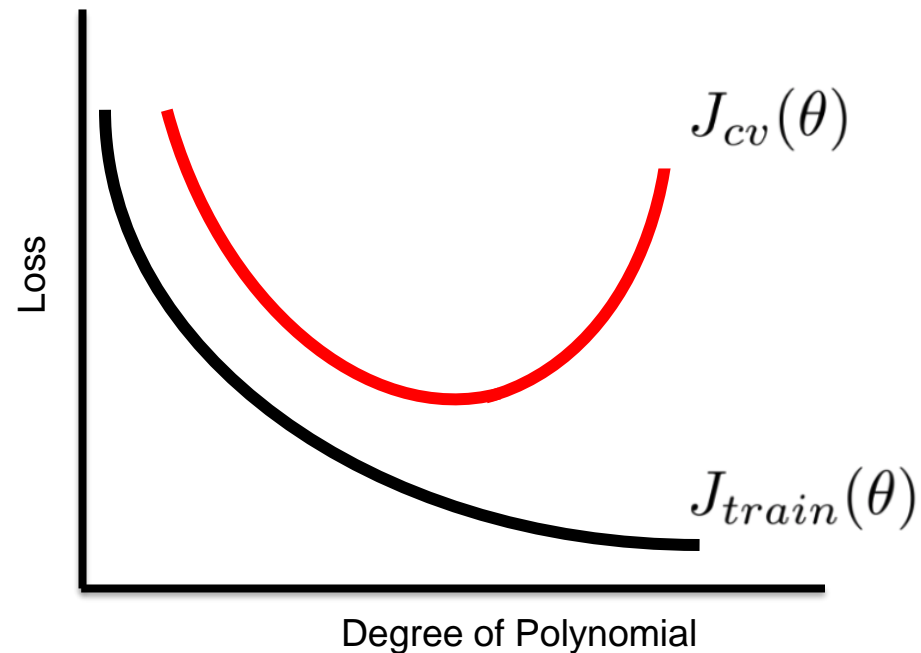
Size (ft)  
Overfit  
High Variance  
Too Complex  
Too little regularization

# Bias / Variance Trade-off

- Training error  $J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$

- Cross-validation error

$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_{\theta}(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$

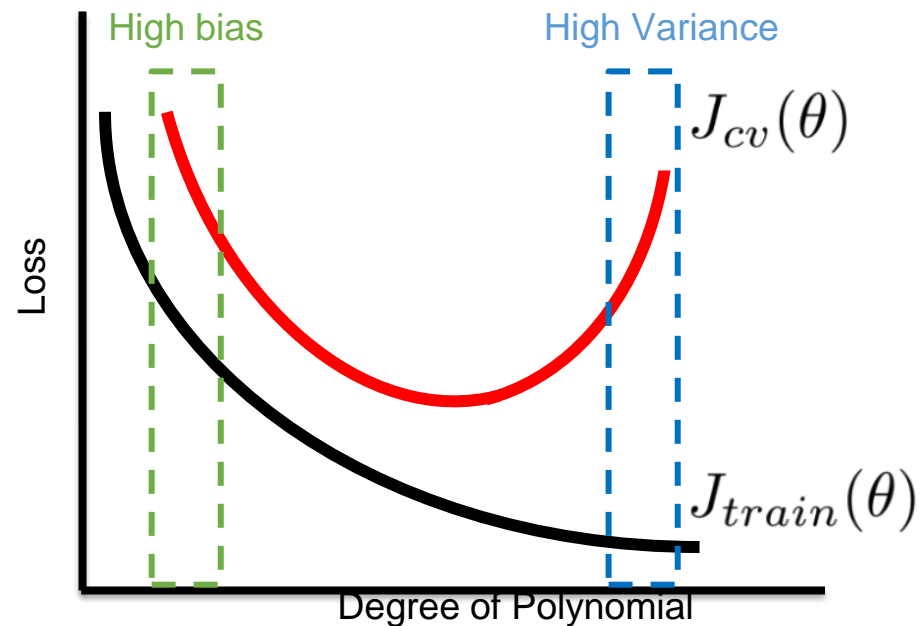


# Bias / Variance Trade-off

- Training error  $J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$

- Cross-validation error

$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_{\theta}(x_{cv}^{(i)}) - y_{cv}^{(i)})^2$$



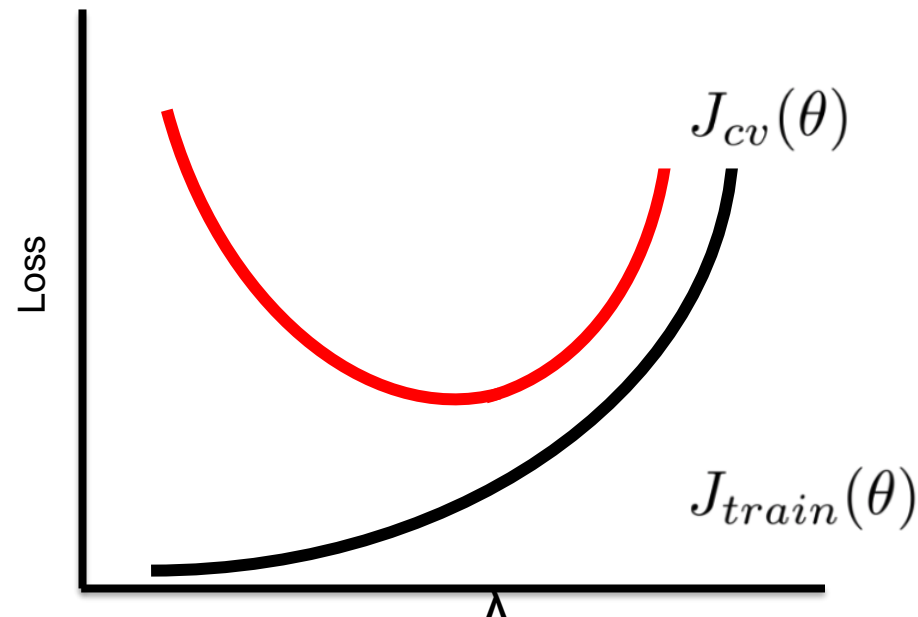
# Bias / Variance Trade-off with Regularization

- Training error

$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^m \theta_j^2$$

- Cross-validation error

$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_{\theta}(x_{cv}^{(i)}) - y_{cv}^{(i)})^2 + \frac{\lambda}{2m_{cv}} \sum_{j=1}^{m_{cv}} \theta_j^2$$



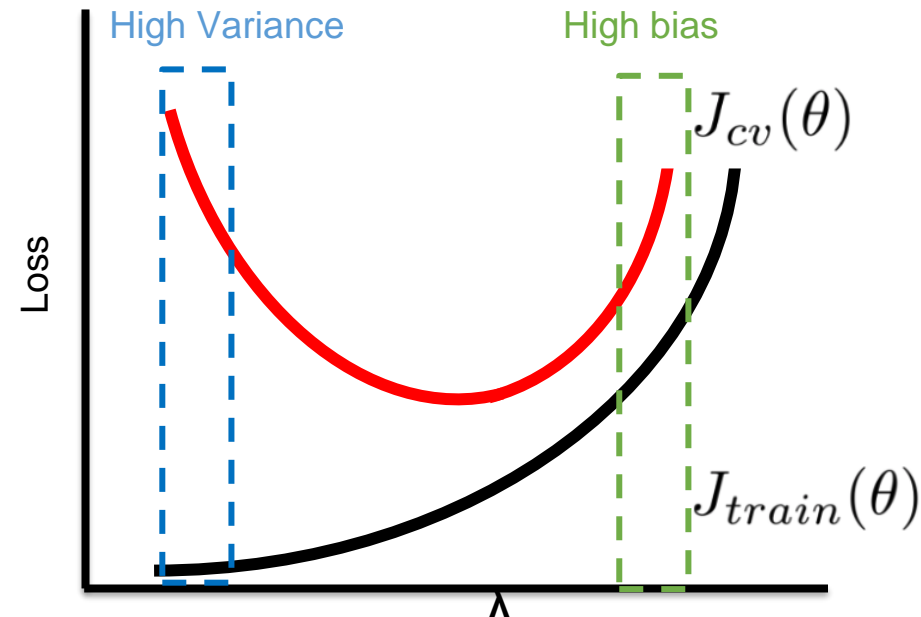
# Bias / Variance Trade-off with Regularization

- Training error

$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2m} \sum_{j=1}^m \theta_j^2$$

- Cross-validation error

$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} (h_{\theta}(x_{cv}^{(i)}) - y_{cv}^{(i)})^2 + \frac{\lambda}{2m_{cv}} \sum_{j=1}^{m_{cv}} \theta_j^2$$





# Problem: Fail to Generalize

- Should we get more data?

# Problem: Fail to Generalize

- Should we get more data?
- Getting more data does not always help

# Problem: Fail to Generalize

- Should we get more data?
- Getting more data does not always help
- How do we know if we should collect more data?

# Things You Can Try

- Get more data
  - When you have **high variance**
- Try different features
  - Adding feature helps fix **high bias**
  - Using smaller sets of feature fix **high variance**
- Try tuning your hyperparameter
  - Decrease regularization when **bias is high**
  - Increase regularization when **variance is high**

# Things You Can Try

- Get more data
  - When you have **high variance**
- Try different features
  - Adding feature helps fix **high bias**
  - Using smaller sets of feature fix **high variance**
- Try tuning your hyperparameter
  - Decrease regularization when **bias is high**
  - Increase regularization when **variance is high**

Analyze your model before you act

The image features a complex network graph with numerous nodes and edges, rendered in a light teal color. The nodes are scattered across the frame, with some forming dense clusters and others being isolated. A semi-transparent white rectangular box is positioned in the upper-middle section of the image, containing the text 'Hyperparameter space' in a bold, black, sans-serif font. The background is a light, neutral gray with a subtle gradient.

# Hyperparameter space

# Manual search



1. experiment
  - 5 hidden layers
  - 1000 neurons
  - Adam
  - Accuracy=0.5



2. experiment
  - 3 hidden layers
  - 1000 neurons
  - Adam
  - Accuracy=0.4



3. experiment
  - 6 hidden layers
  - 1000 neurons
  - Adam
  - Accuracy=0.55

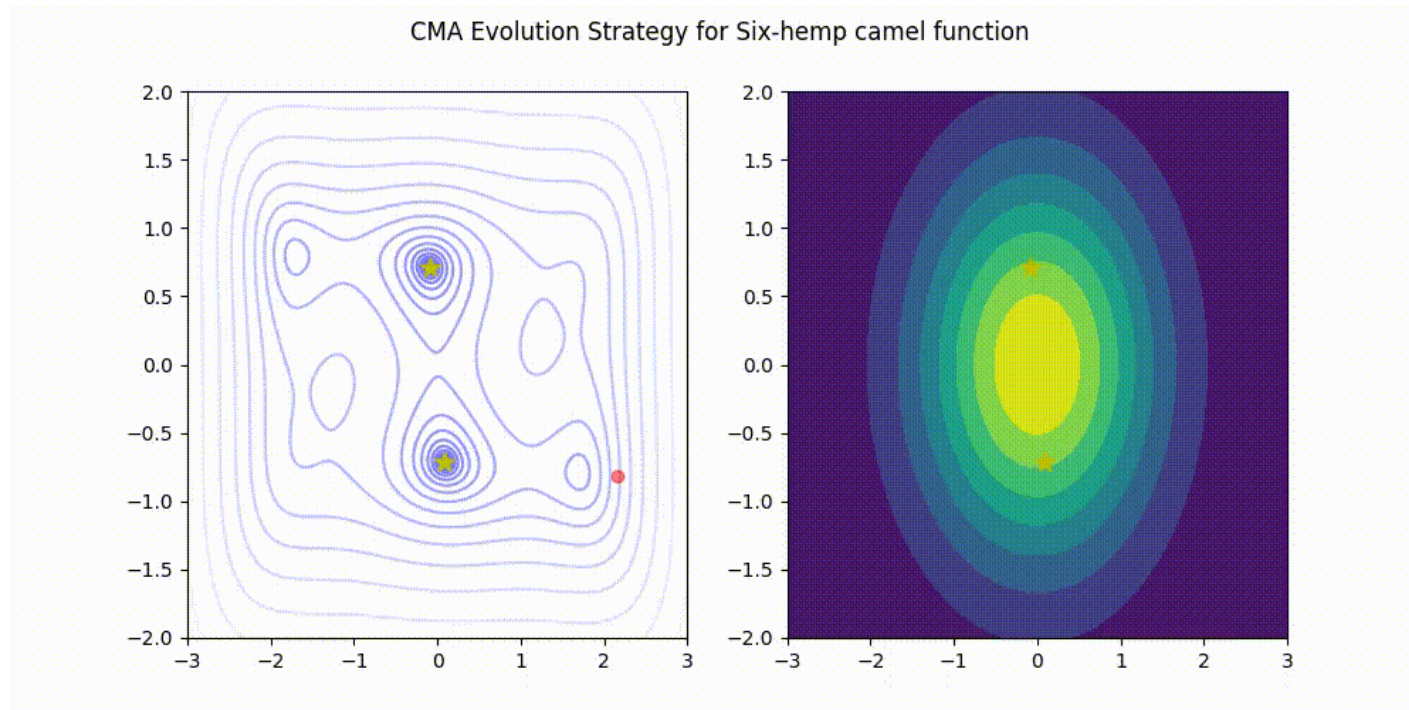


4. experiment
  - 5 hidden layers
  - 2000 neurons
  - Adam
  - Accuracy=0.56

..... ??????

# Bayesian hyperopt / other options

- GP: Gaussian Processes
- CMA-ES: Covariance Matrix Adaptation Evolution Strategy



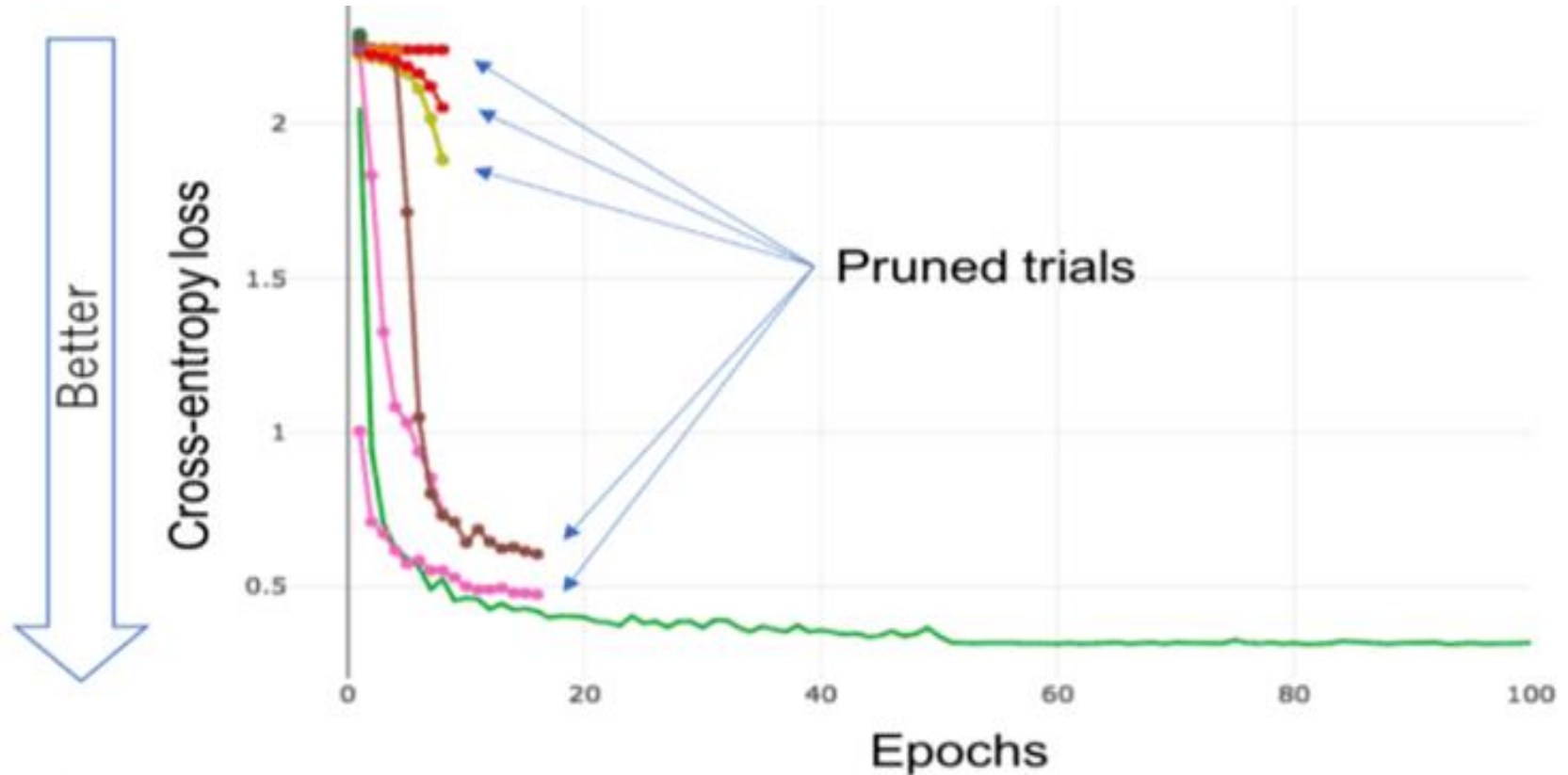


# HyperBand / pruning

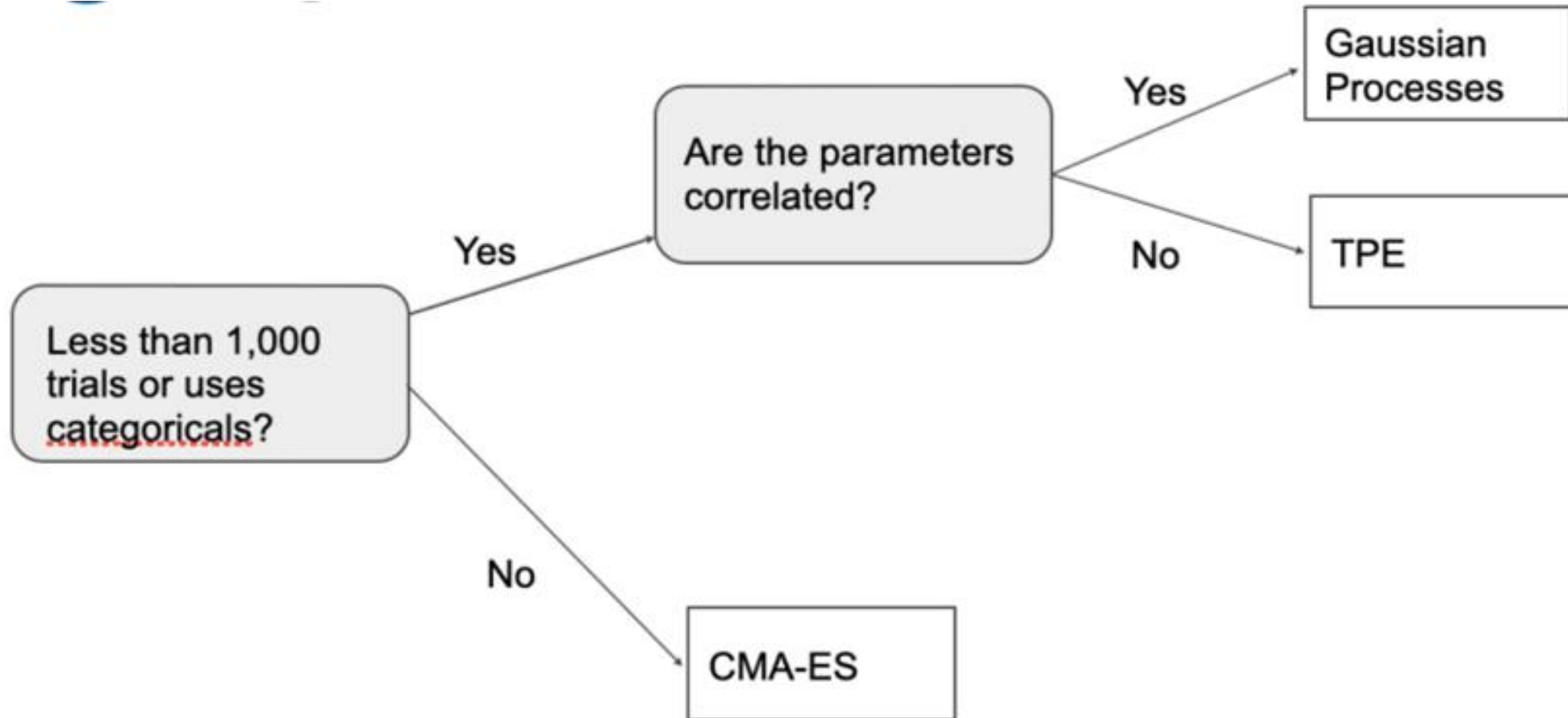
- Idea: don't finish all the trainings!
- the trend can already be seen based on the first epochs
- training for several elements of the hyperparameter space, but only up to 2 epochs
- which of the alternatives could potentially be good?
- further training chosen ones for 2 additional epochs
- ... iterative narrowing of the parameter space

# HyperBand / pruning

- ... iterative narrowing of the parameter space



# Which hyperopt type to choose?



# Distributed hyperopt

```
$ python example.py
[I 2019-05-21 11:16:43,493] Using an existing study with name 'example-study' in
stead of creating a new one.
total [#####.....] 57.17%
this epoch [#####.....] 71.73%
134 iter, 5 epoch / 10 epochs
31.547 iters/sec. Estimated time to finish: 0:00:03.181785.
```

```
$ python example.py
[I 2019-05-21 11:16:42,393] Using an existing study with name 'example-stu
instead of creating a new one.
total [#####.....] 84.91%
this epoch [#####.....] 49.07%
199 iter, 8 epoch / 10 epochs
44.127 iters/sec. Estimated time to finish: 0:00:00.801665.
```

```
$ python example.py
[I 2019-05-21 11:16:43,621] Using an existing study with name 'example-study' in
stead of creating a new one.
total [#####.....] 49.92%
this epoch [#####.....] 99.20%
117 iter, 4 epoch / 10 epochs
27.85 iters/sec. Estimated time to finish: 0:00:04.214611.
```

```
$ python example.py
[I 2019-05-21 11:16:42,230] Using an existing study with name 'example-stu
instead of creating a new one.
total [#####.....] 53.76%
this epoch [#####.....] 37.60%
126 iter, 5 epoch / 10 epochs
26.211 iters/sec. Estimated time to finish: 0:00:04.134763.
```

```
$ python example.py
[I 2019-05-21 11:16:42,729] Using an existing study with name 'example-study' in
stead of creating a new one.
total [#####.....] 51.63%
this epoch [#####.....] 16.27%
121 iter, 5 epoch / 10 epochs
25.56 iters/sec. Estimated time to finish: 0:00:04.435707.
```

```
$ python example.py
[I 2019-05-21 11:16:42,022] Using an existing study with name 'example-stu
instead of creating a new one.
total [#####.....] 56.32%
this epoch [#####.....] 63.20%
132 iter, 5 epoch / 10 epochs
26.968 iters/sec. Estimated time to finish: 0:00:03.796150.
```

A network graph visualization with nodes and edges, overlaid with a white text box. The nodes are small circles in various shades of blue and green, connected by thin lines. The graph is spread across the width of the slide, with a central horizontal band where the text is located.

# Analysis & visualization of hyperopt results

# Hyperopt final result

```
[I 2020-09-11 17:26:09,841] Trial 97 pruned.  
[I 2020-09-11 17:26:10,217] Trial 98 pruned.  
[I 2020-09-11 17:26:10,790] Trial 99 pruned.  
Study statistics:  
  Number of finished trials: 100  
  Number of pruned trials: 67  
  Number of complete trials: 33  
Best trial:  
Value: 0.95390625  
Params:  
  n_layers: 1  
  n_units_l0: 115  
  dropout_l0: 0.40951215992211487  
  optimizer: Adam  
  lr: 0.021951376926653072
```



Source: [https://github.com/optuna/optuna/blob/master/examples/pytorch\\_simple.py](https://github.com/optuna/optuna/blob/master/examples/pytorch_simple.py)

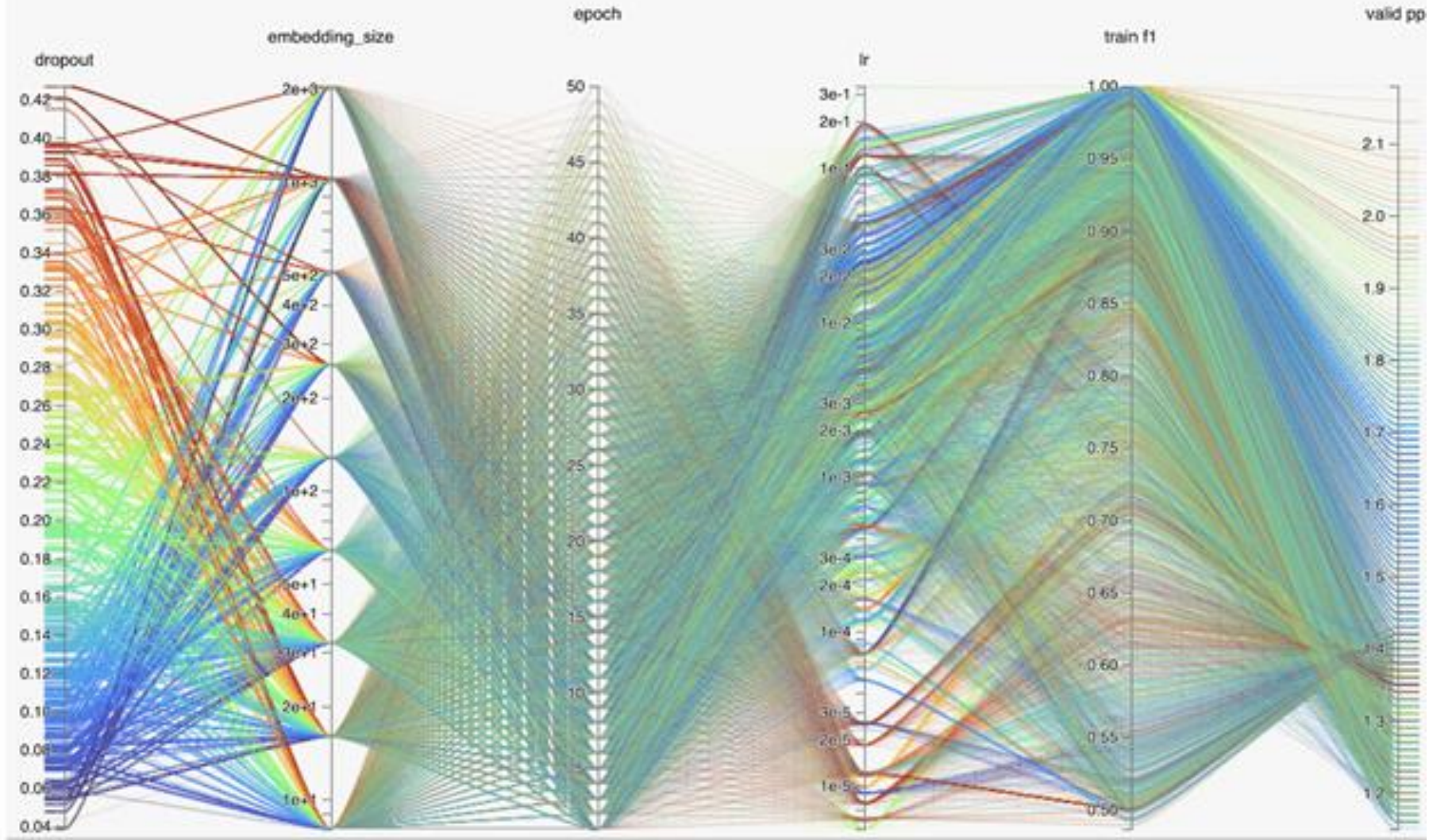
Source: <https://optuna.org>





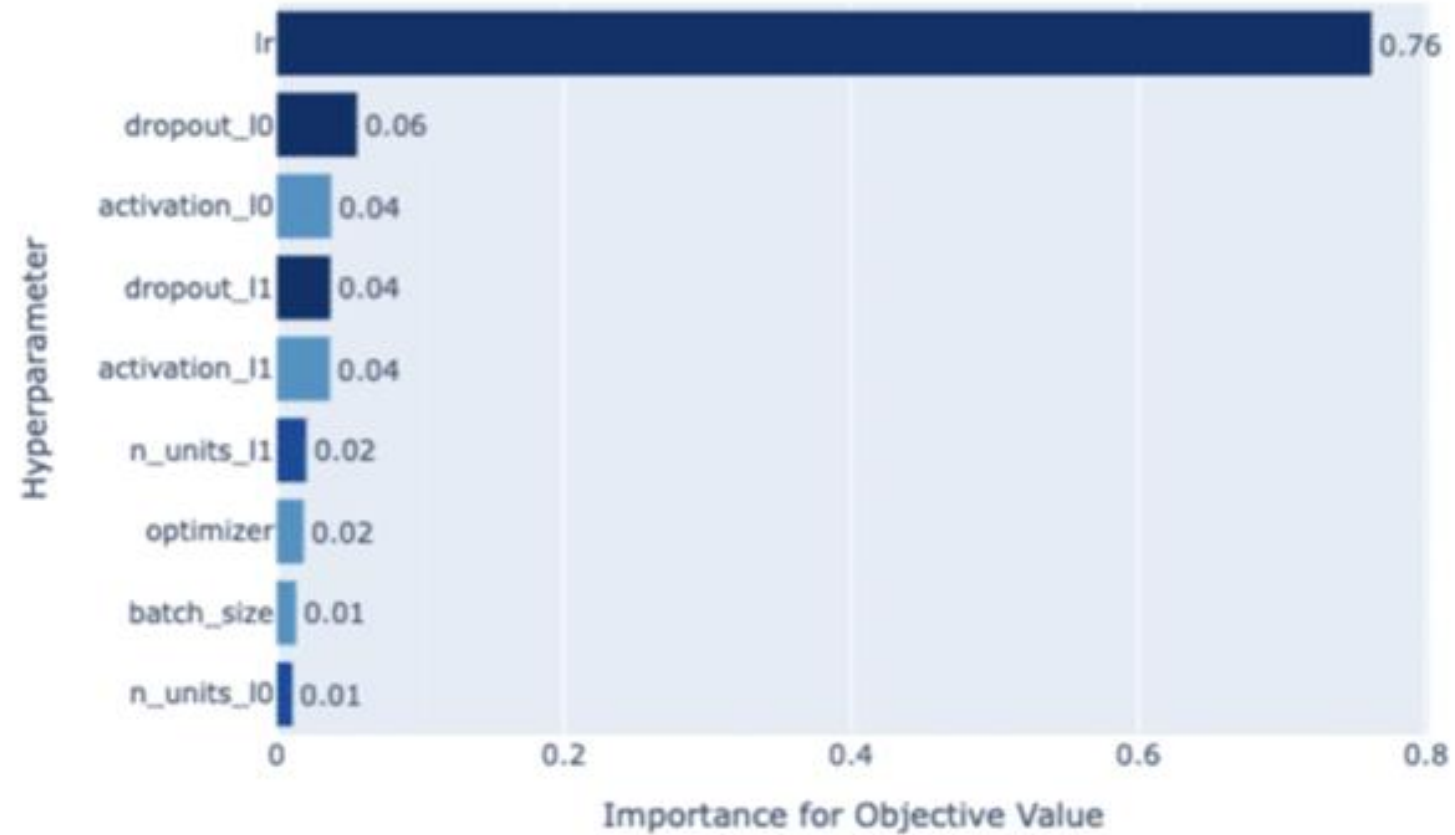
Restore Keep Exclude Export Help

Selected: 6968/6968 (100%)



# Which hyperparameter is the most important?

Hyperparameter Importances





# Weights and Biases

- Works fine in colab
- Can be installed locally using docker
  - <https://hub.docker.com/r/wandb/local>
  - <https://docs.wandb.ai/guides/launch/docker>
- Example for the visual result
  - <https://wandb.ai/wandb/examples-keras-cnn-fashion/sweeps/us0ifmrf>

# Weights and Biases - Sweeps

- <https://docs.wandb.ai/guides/sweeps>

# HyperOpt frameworks

- <https://github.com/maxpumperla/hyperas>
- <https://github.com/keras-team/keras-tuner>
- <https://github.com/autonomio/talos>
- <https://github.com/sherpa-ai/sherpa>
  
- <https://github.com/optuna/optuna>
- <https://github.com/hyperopt/hyperopt>
- <https://github.com/Avsecz/kopt>
- <https://github.com/tobegit3hub/advisor>
- <https://github.com/Alworx-Labs/chocolate>
- <https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras/>

# General hyperopt code

```
import <hyper>

def objective(trial):
    <DNN definition, with hyperparams>

    return evaluation_score

study = <hyper>.create_space()
study.optimize(objective,
               search_algorithm,
               n_trials, ...)
```

A network graph with nodes and edges, overlaid with a white rectangular box containing the text 'References'. The graph consists of numerous nodes connected by thin lines, forming a complex web. The nodes are colored in shades of blue and green. The white box is centered horizontally and contains the word 'References' in a black, sans-serif font.

# References

# References

- <https://optuna.org>
- [https://keras.io/keras\\_tuner/](https://keras.io/keras_tuner/)
- <https://wandb.ai>

Please, don't forget  
to send feedback:

<https://bit.ly/bme-dl>



# Thank you for your attention

Dr. Mohammed Salah Al-Radhi  
[malradhi@tmit.bme.hu](mailto:malradhi@tmit.bme.hu)

(slides by: Dr. Tamás Gábor Csapó)

18 September 2024

