Hyperparameters

Machine Learning Operations

Nicki Skafte Detlefsen,

Postdoc

DTU Compute

Why do we optimize hyperparameters

https://paperswithcode.com/

Many reasons

1. Get a working model

2. Get a more robust model

3. Get better performance

4. Something else…

"We are all SOTA suckers"

    - Ole Winther

Nicki Skafte Detlefsen

What is considered a hyperparameter?

- Everything that effects training:
  - Model architechture
  - Optimizer structure

- Question: Is the following a hyperparameter?
  - Dataset
  - Seed

Nicki Skafte Detlefsen
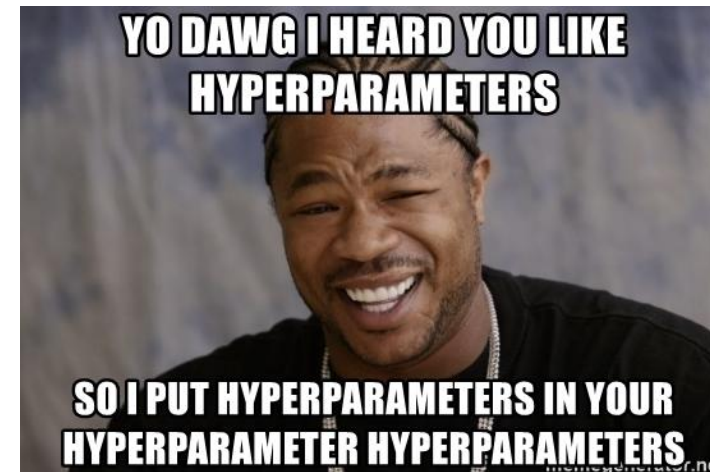
Detour: What should we optimize?

Regardless of chosen model, hyperparameter, search strategy etc. it all depends that we can define some **Metric** for which we can determine if a set of hyperparameters h_1 is better than h_2
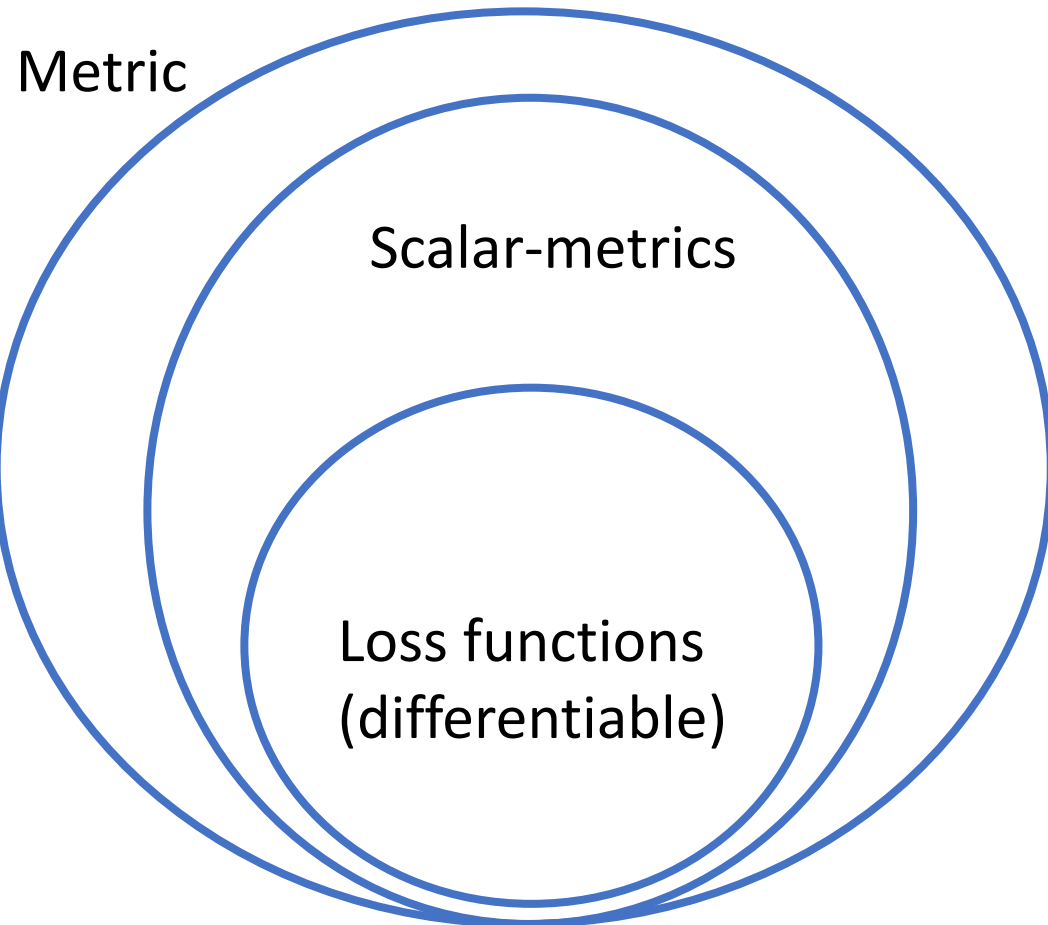
In general we will say that

$$h_1 \text{ is better than } h_2$$

If it holds for our **Metric M** that

$$M(h_1) < M(h_2)$$

What is a metric?

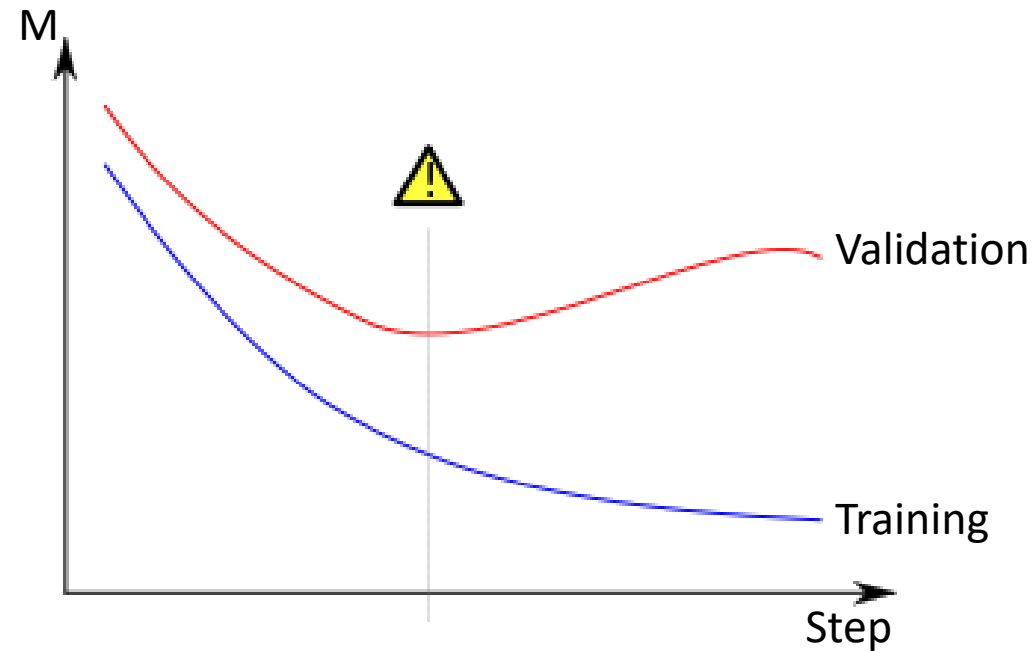MLOps

Any quantity that measures "goodness of fit"

Metric

Scalar-metrics

Loss functions
(differentiable)

Confusion Matrix:

Do you recognize this figure?

One detour deeper: Generalization error

The generalization error/expected loss/risk is given by

$$I[f] = \int_{X \times Y} M(f(x), y)p(x, y)\,dxdy$$

where f is some function $f: X \to Y$, M denotes the evaluation Metric and $p(x, y)$ is the joint probablility distribution between x and y

We cannot optimize this directly because $p(x, y)$ is unknown and even if we knew it, the integral would be intractable.

Generalization error

We can calculate the empirical error

$$I_n[f] = \frac{1}{n} \sum_{i=1}^{n} M(f(x_i), y_i)$$

Which measured the "error" that our function f does on n datapoints measured by metric M
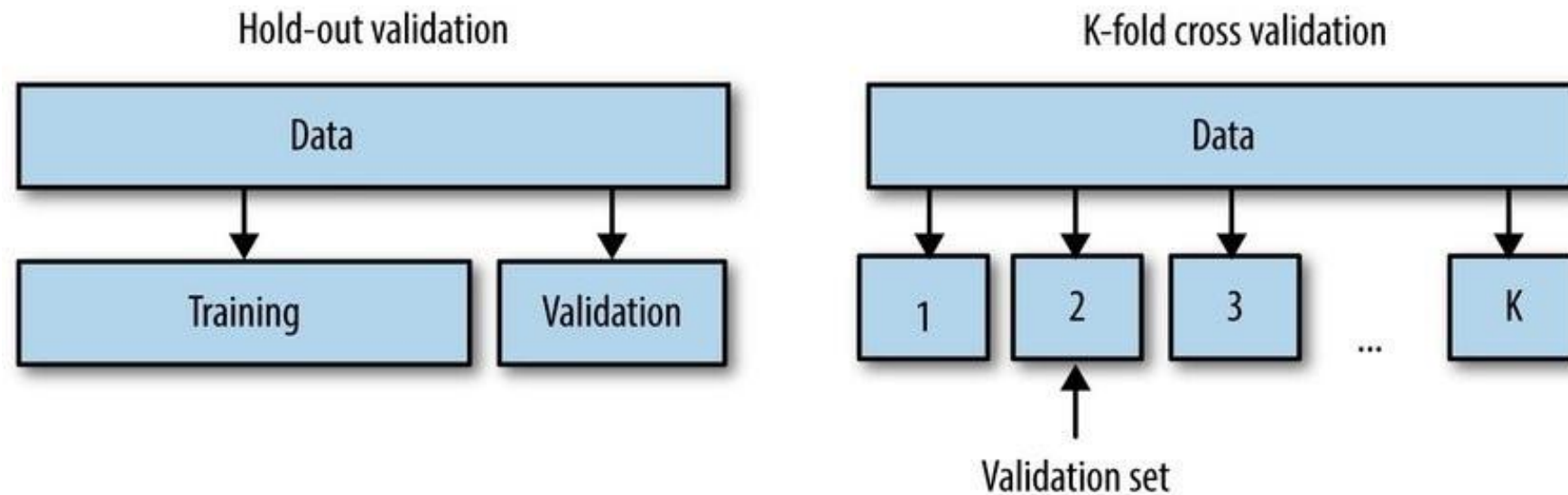
An algorithm is said to **generalize** if
$$\lim_{n \to \infty} I[f] - I_n[f] = 0$$

however, this does not solve our problem because $I[f]$ is still unknown

Cross-validation

Cross-validation in a nutshell:

We take some of our data away which we can use to estimate the true generalized loss $I[f]$

Hold-out validation

K-fold cross validation

Question: Which one do we use in deep learning and why?
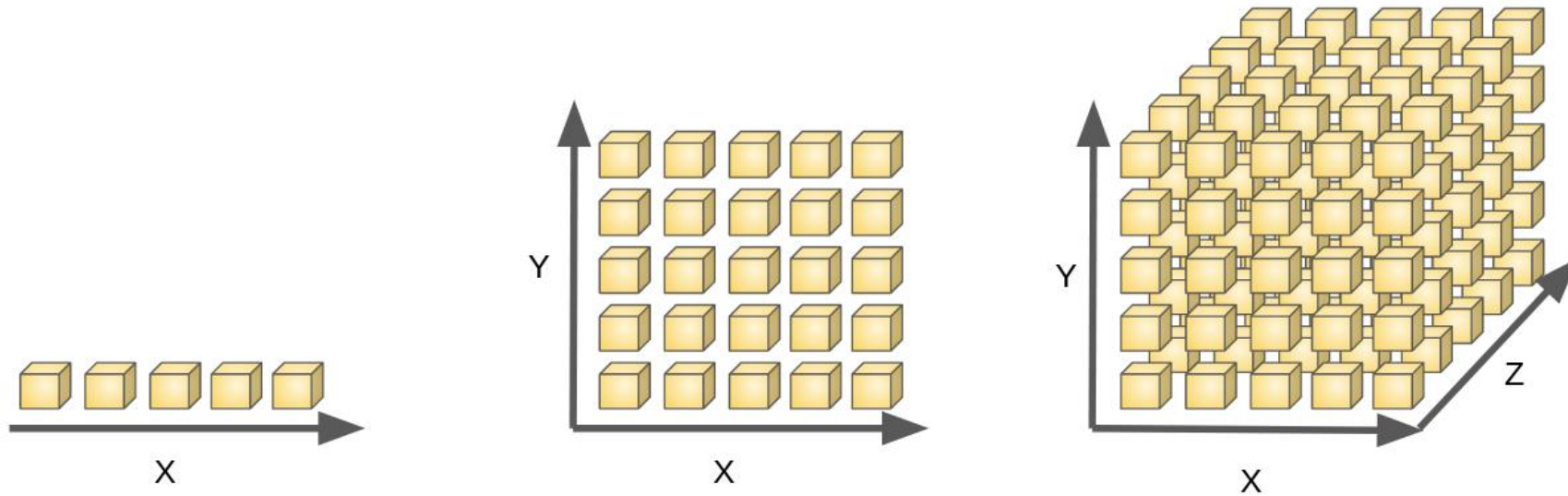
Back to topic

We now assume that we have

1. Some metric we trust

2. Some way to measure the generalization error (validation set)

3. Some hyperparameters that we want to optimize

How should we investigate our hyperparameter space?

What about doing grid search?

Curse of dimensionality anyone?

The number of combinations grows exponentially in the number of parameters we try to optimize.
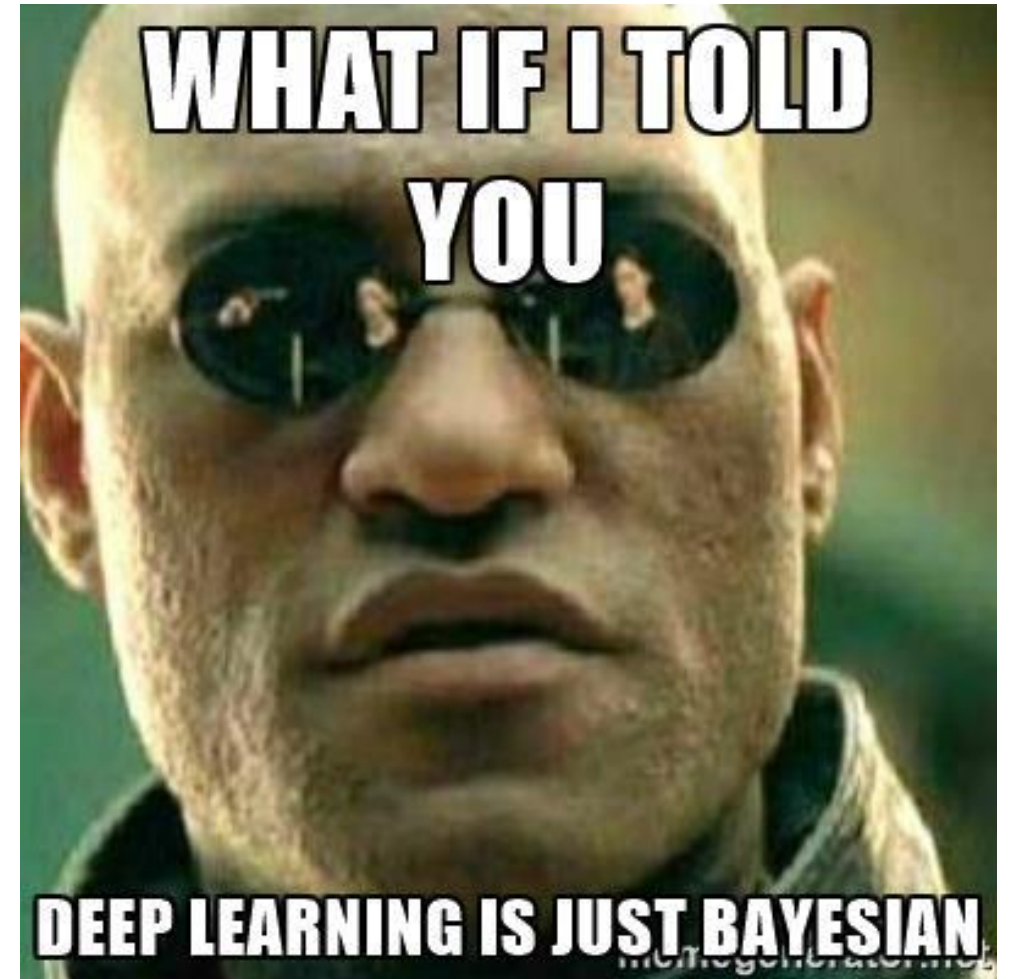
Baysian to the rescue

Baysian optimization tries to find solution to the problems of the form

$$\max_{x \in A} f(x)$$

Where A is known but f may be arbitarily complex

Hyperparameter optimization of the neural networks fit this formulation perfectly meaning that the problem is already solved...

WHAT IF I TOLD YOU

DEEP LEARNING IS JUST BAYESIAN

Hyperparameter framework

https://optuna.readthedocs.io/en/stable/

Optuna give easy acess to two tricks for hyperparameter optimization

1. Sampling
2. Pruning

Trick 1: Sampler

Samples hyperparameters in a smart fashion that takes into account the function values we have already seen

Optuna as default uses:

- NSGA-II
  - Evolutionary algorithm
    - https://ieeexplore.ieee.org/document/996017

Step One: Generate the initial population of individuals randomly. (First generation)

Step Two: Repeat the following regenerational steps until termination:

1. Evaluate the fitness of each individual in the population (time limit, sufficient fitness achieved, etc.)
2. Select the fittest individuals for reproduction. (Parents)
3. Breed new individuals through crossover and mutation operations to give birth to offspring.
4. Replace the least-fit individuals of the population with new individuals.

Source: http://haikufactory.com/files/bayopt.pdf

Nicki Skafte Detlefsen
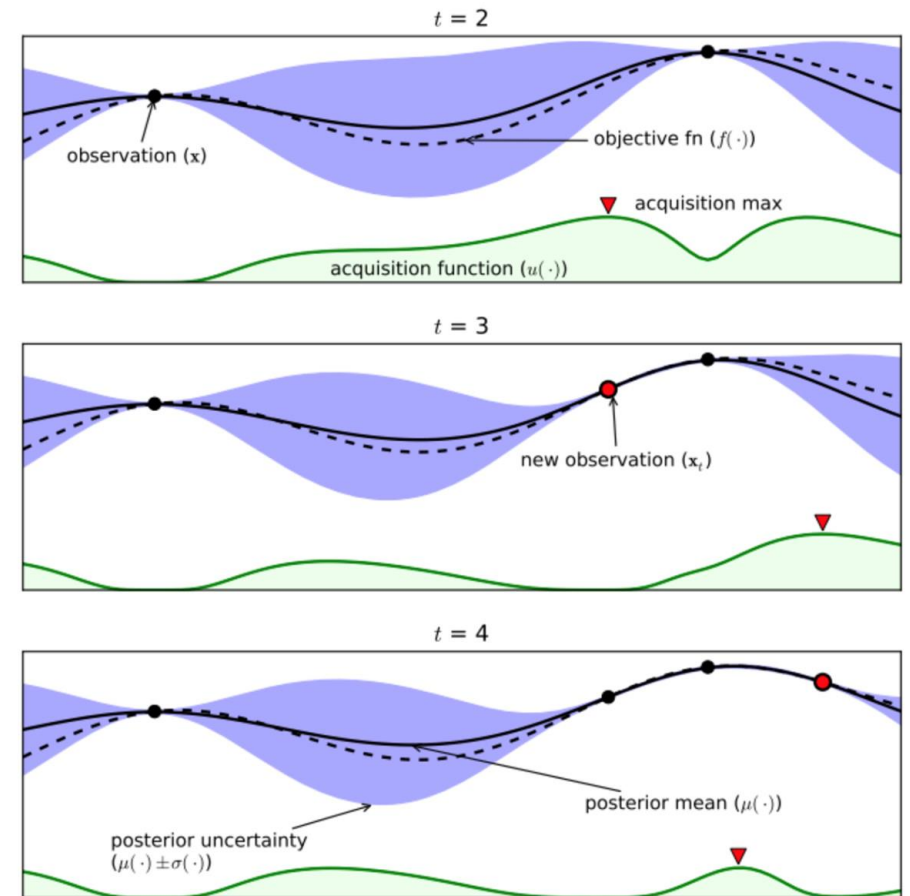
Trick 2: Pruning

Cut away trials that look unpromising (often based on the median performance)
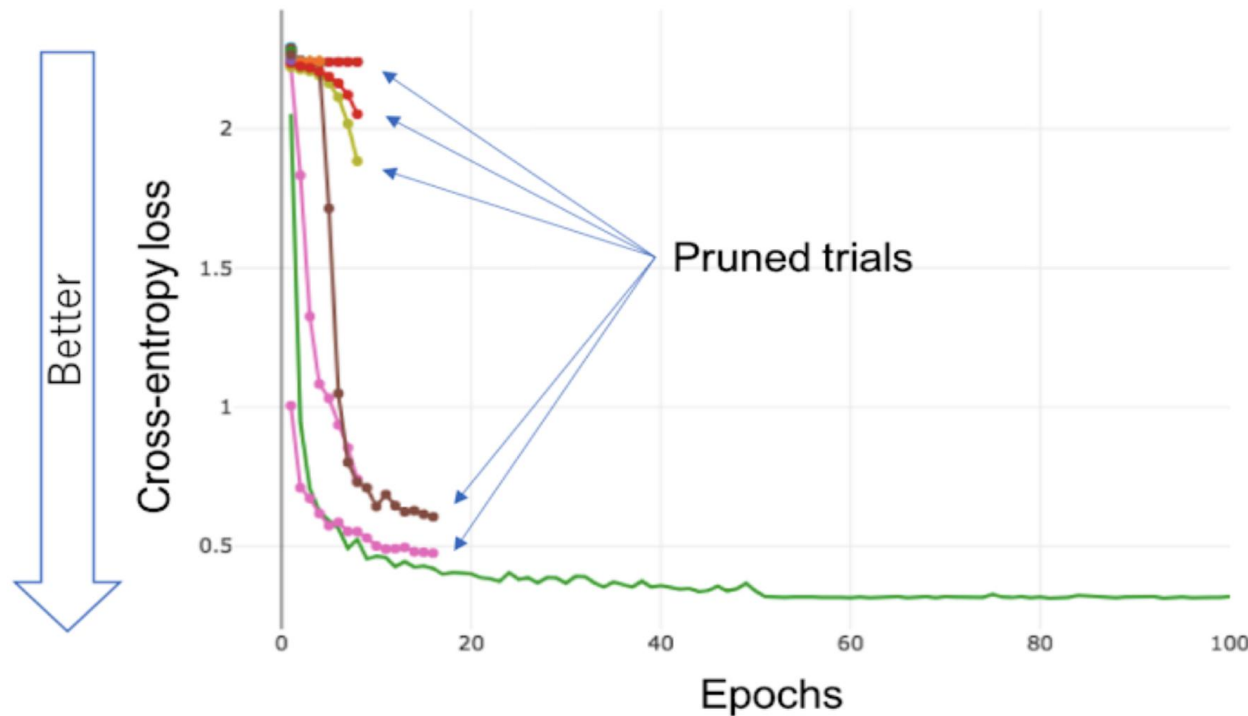
Be careful with over-pruning as you may lose some performance on the floor

Meme of the day

Nicki Skafte Detlefsen