Neuronale Netze (SS 2002), 13.5.

The whole training pipeline for a FNN:

- **Get the data** from a good friend, a benchmark collection, your real-life problem, ...

  Yields a set $\{\vec{x}^i, f(\vec{x}^i) + \eta_i \mid i = 1, \ldots, P\}$ where η_i is noise and f is an unknown function which is to be learned from the data.

- **Data preprocessing**: Data are to be presented appropriately.

  Idea: make the problem as easy as possible, involve all known invariances, all available prior knowledge

  Often: problem dependent, takes a lot of time, crucial for success

  $\rightarrow$ best to ask an expert in the respective application area

- **Architecture selection:** Choose an appropriate neural architecture.

  Idea: should have the right number of free parameters, not too small (f should be representable) and not too large (the noise η_i should not be learned).

  Often: take multilayer architecture with 1-2 hidden layers, around 5/10 hidden neurons per layer; simply try, which architecture is best (we'll get the method: crossvalidation)

  Question: Approximation capability of FNNs and bounds on the recources

- **Network training:** backpropagation and further tricks (we'll get some)

  Idea: make the training error small, possibly additional goals

  Question: Efficiency of training, guarantees for convergence

- **Interpretation of training:** what are the outputs for your training problem, is the underlying function f learned?

  Idea: the network should perform as good as possible on unknown input data, it should now represent f accurately

  Question: Guranatee of the generalization ability?

Preprocessing:

- encode symbolic attributes unary/in an ordered fashion
- scale real valued inputs such that their range corresponds to their importance
- complete missing attributes e.g. with default values
- have a look at the outputs, too!
- for time-series-prediction:
  - choose a time window
  - choose additional global attributes
  - make sure that your time series is stationary, e.g. building differences
- image classification:

  line extraction, smoothing, wavelet, Fourier transform, filters, segmentation, scaling, . . .

Always look at the data before training!