Thoughts on Distributed Computing

Peter Wittenburg
Max Planck Society
The Problem

Network size remains a myth (avoid overspecification)

Training Sets get large
(ImageNet: 16 Mio Images)
I/O Problem

Category Sets get large

Neural Networks get large
N layers, M nodes per neuron
CPU & Communication Problem
Training / Classification

Typical Back Propagation Example

Training (extreme load)
- use large training set to calculate $w$
- for each input pattern calculate output
- calculate classification error
- based on error calculate $\Delta w$ backwards
- typical:
  - nr. of iterations
  - change network model

Classification (less load)
- for each pattern to classify calculate output
Approaches

Using parallelism (distribution on compute nodes)
- data parallelisation
- model parallelisation
- task parallelisation

Using Edge Computing
("CEO" Accenture Germany: ML to come everywhere)
- classification only (?)
- task parallelisation
- ?

Recurrent Networks
- working with real time streams & updating model
Parallelisation

data parallelisation
(high communication effort to serve parameters)

model parallelisation
(high communication effort dependent on model type)
Task Decomposition

many „small“ detectors
- normal pattern recognition did not work
- reducing training complexity
- reducing detector complexity
- introducing phased training
- but
  - creating a complex lattice
  - need to understand the problem (features, etc.)
Recurrent Neural Networks

**adding backward weights**
- adding complexity
- adding a memory
- but: stability issue with feedback loops
- we used it for speech recognition stepping away from HMM
- could use it for any other realtime streams