Extrapolating Learning Curves of Deep Neural Networks

Tobias Domhan, Jost Tobias Springenberg, Frank Hutter

Freiburg University

{domhant,springj,fh}@informatik.uni-freiburg.de

… in 30 sec

- **Deep networks** critically depend on **hyperparameters**, but training is **expensive**
- To automate a heuristic that experts use, we built a **probabilistic model** to **forecast the asymptotic accuracy** of a given parameter setting and stop all but the most promising runs
- Simulation resulted in a **2.7-fold reduction of overall runtime**

Motivation

- It takes very few SGD iterations for a human expert to tell good from bad parameter settings
- Yet in hyperparameter optimization every setting is run to the very end
- Automating the prediction of performance **could save a lot of time** and **speed up preliminary evaluations** during development

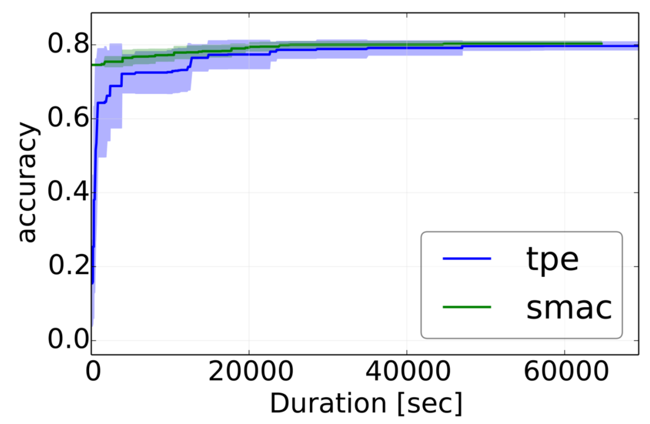
Model Search

- **Search over structure and hyperparameters** of deep networks:
  - **81 parameters in total**, namely 9 network parameters and 12 parameters for each of up to 6 layers
  - Neural network software: Caffe [Jia 2013]
- Bayesian optimization methods:

SMAC (Sequential Model-based algorithm configuration) is based on random forests and can handle continuous, discrete and conditional hyperparameters.
[Hutter, Hoos, and Leyton-Brown, 2011]

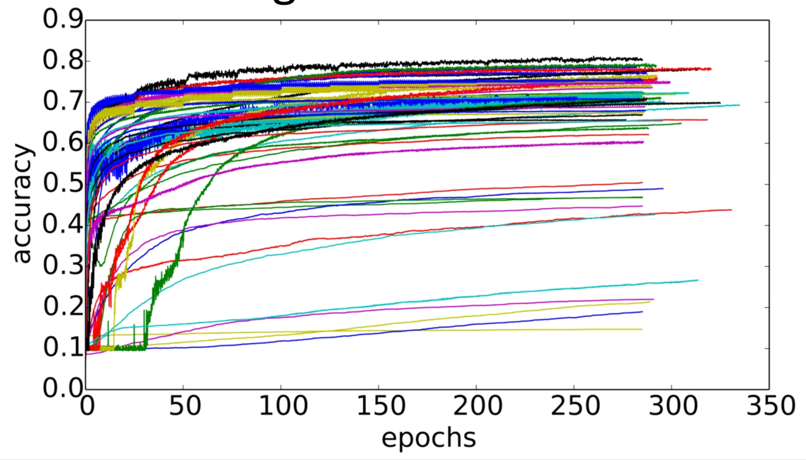
TPE (Tree Parzen Estimator) is based on Gaussian Mixture Models. Supports conditional, continuous and discrete parameters and also priors over them.
[Bergstra, Bardenet, Bengio, and Kégl, 2011]

- 5 runs of both SMAC and TPE
- Evaluated a total of **800 networks**
- Dataset: k-means features extracted from CIFAR10
  [Krizhevsky 2009; Coates 2011]



Learning curves

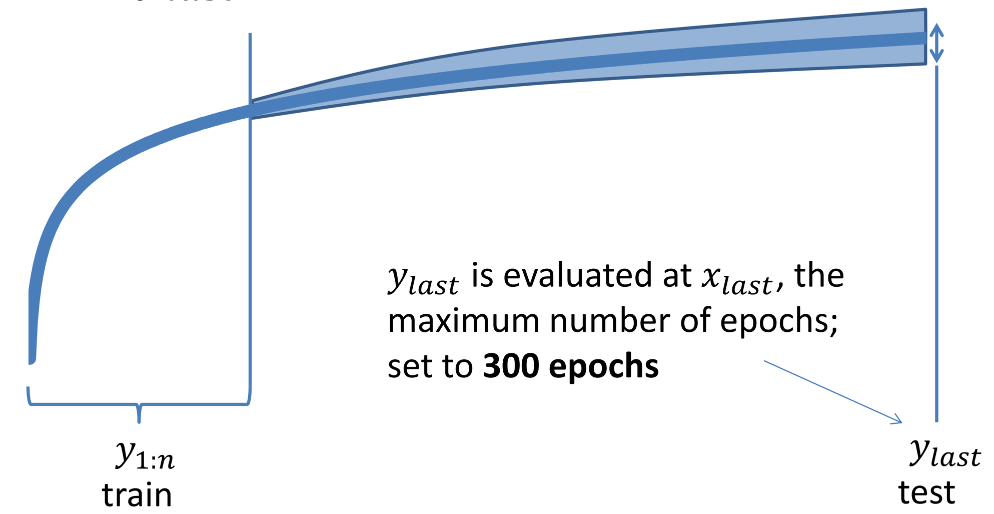
- Random subset of learning curves:



Extrapolation

Problem definition

- Given data points $y_{1:n}$ we would like to **forecast the future performance** y_{last} **probabilistically**



Approach

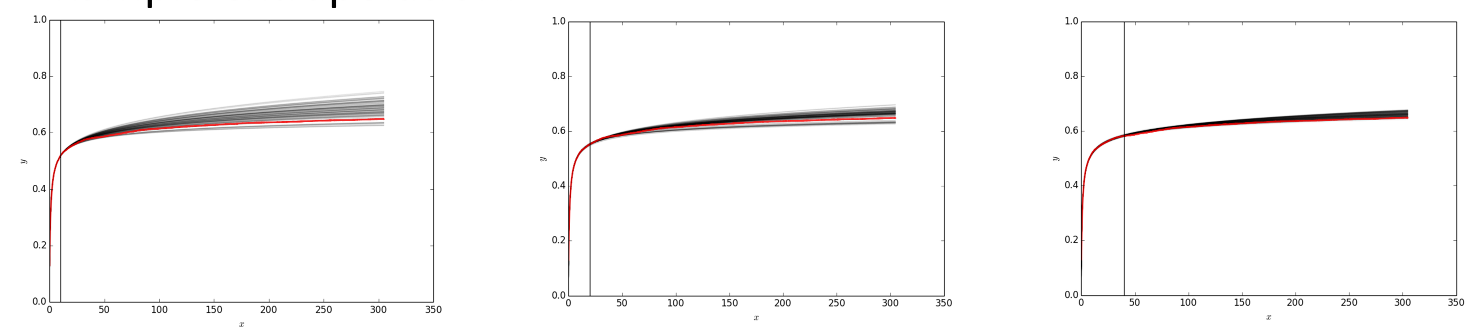
- Selected $k = 10$ **parametric model families** that roughly match learning curves' shape (typically increasing, saturating functions)

pow$_3$: $c - a\,x^{-\alpha}$	pow$_4$: $c - (a\,x + b)^{-\alpha}$
Exp$_4$: $c \;-\; e^{-ax^{\alpha}+b}$	Janoschek: $\alpha - (\alpha - \beta)e^{-k\,x^{\delta}}$
DR-hill: $\frac{t\,x^{\eta}}{\kappa^{\eta}+x^{\eta}}$	MMF: $\alpha - \frac{\alpha - \beta}{1.+(\kappa\,x)^{\delta}}$
vap: $e^{(a+\frac{b}{x}+c\ \log x)}$	loglog linear: $\log(a\log(x) + b)$
ilog$_2$: $c \;-\; \frac{a}{\log x}$	Weibull: $\alpha - (\alpha - \beta)\,e^{-(\kappa x)^{\delta}}$

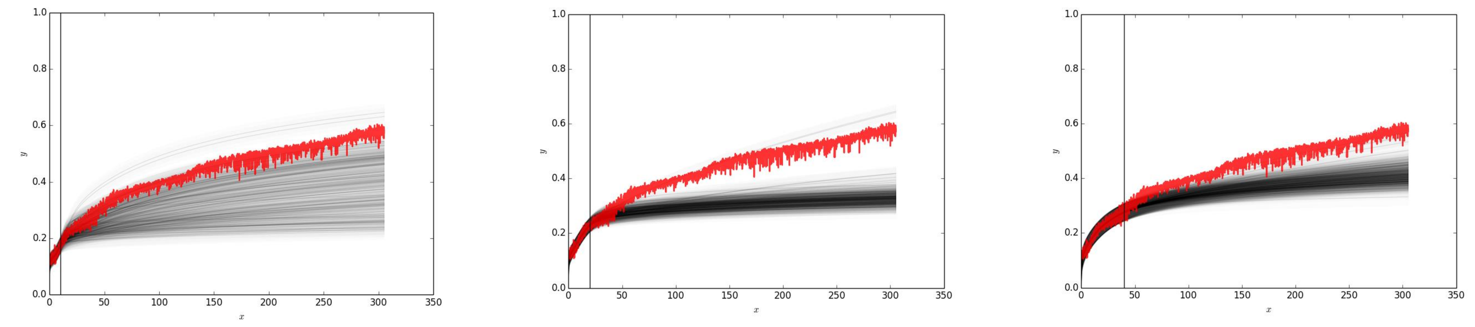
- Representative power increased by **convex combination of individual models**:
  - $f(x) = \sum_{i=1}^{k} w_i f_i(x|\boldsymbol{\theta}_i) + \epsilon$ with $\epsilon \sim N(0, \sigma^2)$ and $\sum_{i=1}^{k} w_i = 1$
- Model uncertainty captured by **MCMC**
  - The prior encoded monotonicity assumption of each of the models
  - We obtained $S = 100000$ samples from 100 parallel chains of length 1500 with a burn-in of 500
  - Let $\boldsymbol{\xi}$ be the model's parameters $(w_1, \dots, w_k, \boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_k, \sigma^2)$
- **Probability of improving over current best** parameter setting:
  - $P(y_{last} \geq y_{best} \mid y_{1:n}) \approx \frac{1}{S}\sum_{s=1}^{S} P(y_{last} \geq y_{best} \mid \boldsymbol{\xi}^{(s)}, y_{1:n})$

Experiments

- Example extrapolation:



- Example of model being misled by unusual shape of the learning curve:



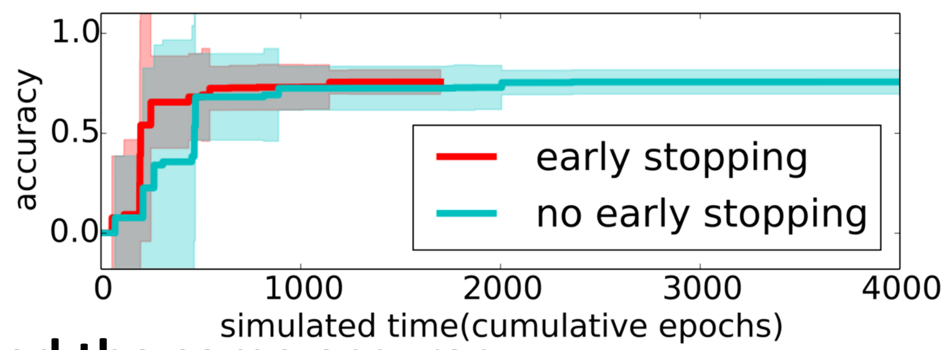
- Quality of predictions:
  - RMSE of residual $E[y_m] - y_{last}$:

% train	10%	30%	50%	70%	90%
RMSE	0.082	0.046	0.026	0.010	0.011

  - y_{last} in/over/under 90% interval:

% train	10%	30%	50%	70%	90%
y_{last} in	42.54 %	48.51%	61.94 %	80.45 %	91.04 %
y_{last} over	12.69 %	9.70 %	8.96 %	6.77 %	6.72%
y_{last} under	44.77 %	41.79 %	29.10 %	12.78 %	2.24 %

- Model tends to be overconfident based on little data, but rarely underpredict
- Simulated early stopping in optimization
  - Replayed all 800 runs
  - Stopped a run when probability of improving over current best got too small: $P(y_{last} \geq y_{best} \mid y_{1:n})$ < 1%



  - **Reached the same accuracy**
  - **2.7-fold speedup**

Ongoing/Future Work

- Use early stopping in model search
- Control early stopping via Bayesian optimization