Towards Automated Deep Learning: Efficient Joint Neural Architecture and Hyperparameter Search

Arber Zela Aaron Klein Stefan Falkner Frank Hutter

University of Freiburg

{zelaa, kleinaa, sfalkner, fh}@cs.uni-freiburg.de

In a Nutshell

- Optimizing hyperparameters and neural network architectures **separately** may be suboptimal due to interactions between them
  - We optimize a **joint** 17-dimensional architecture and hyperparameter space and achieve competitive results for just 3 hours of training
- Performance after short and long training budgets only **correlates weakly**
  - But correlation with intermediate budgets is much higher
  - We use BOHB (Bayesian Optimization Hyperband) [Falkner et al. 2018] to **incrementally increase budgets** during optimization

Related Work

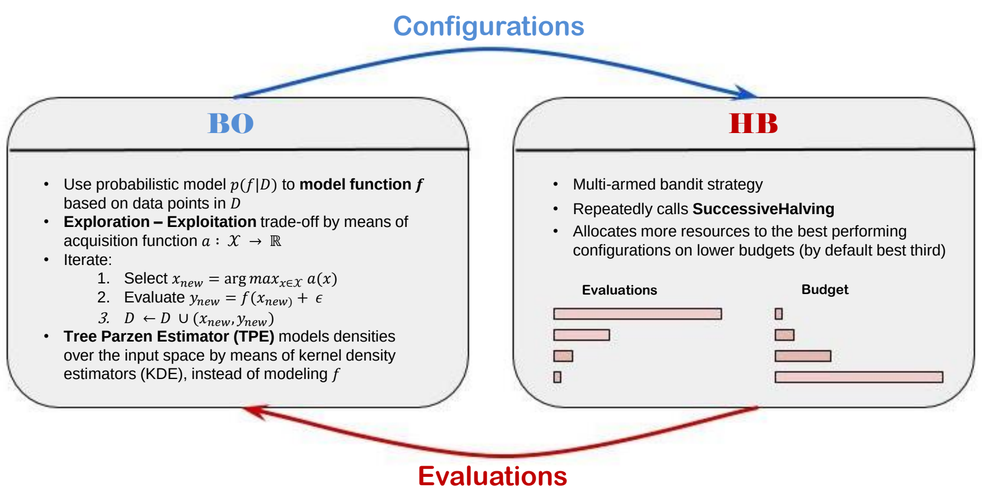
Many recent works on neural architecture search, but all of them use two-step optimization (first architecture, then hyperparameters). E.g.:

- **Reinforcement Learning** [Zoph et al. 2018]: Train a controller RNN with PPO to sample string encoding of the architecture
- **Neuro-evolution** [Liu et al. 2018a]: mutate population of models and add to the population the best offsprings (w.r.t. validation error)
- **Sequential model-based optimization** [Liu et al. 2017]: learn surrogate model and sample more efficient architectures
- **Gradient-based** [Liu et al. 2018b]: parameterize network architecture by creating mixed operations and optimize using gradient descent

Original Bayesian optimization NAS papers already used joint optimization:

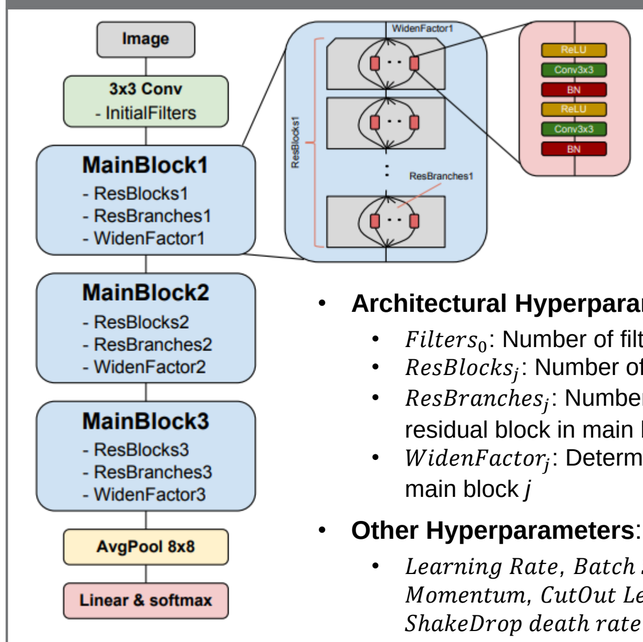
- **Bayesian optimization** [Bergstra et al. 2013, Domhan et al. 2015, Mendoza et al. 2016]: achieved state-of-the-art on several datasets using tree-based models

BOHB



- BOHB samples from a **learned probabilistic model** instead of randomly
- It uses a multivariate KDE to better model interactions between parameters

Search Space



Baseline: Multiple-branch ResNet.

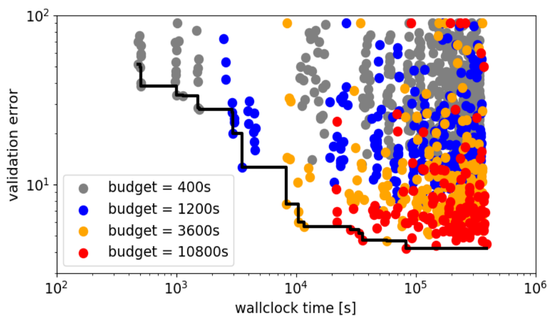
Regularization:
- Shake-Shake (if > 2 branches)
- ShakeDrop (if 2 branches)
- CutOut
- MixUp

- **Architectural Hyperparameters**:
  - $Filters_0$: Number of filters for the first convolution
  - $ResBlocks_j$: Number of residual blocks for main block j
  - $ResBranches_j$: Number of residual branches for each residual block in main block j
  - $WidenFactor_j$: Determines the number of filters after main block j
- **Other Hyperparameters**:
  - $Learning\ Rate,\ Batch\ Size,\ L_2\ regularization,\ Momentum,\ CutOut\ Length,\ MixUp\ \alpha,\ ShakeDrop\ death\ rate$

Experiments

Results

- **Limited budgets** (training time): 400s, 1200s, 1h and 3h



Model	Param (M)	Test Error
ResNet-18	11.2	3.34 ± 0.11
Shake-Shake 26 2x32d	2.9	3.91 ± 0.09
Shake-Shake 26 2x64d	11.7	3.38 ± 0.07
Shake-Shake 26 2x96d	26.2	4.42 ± 0.06
Ours	27.6	3.18 ± 0.16

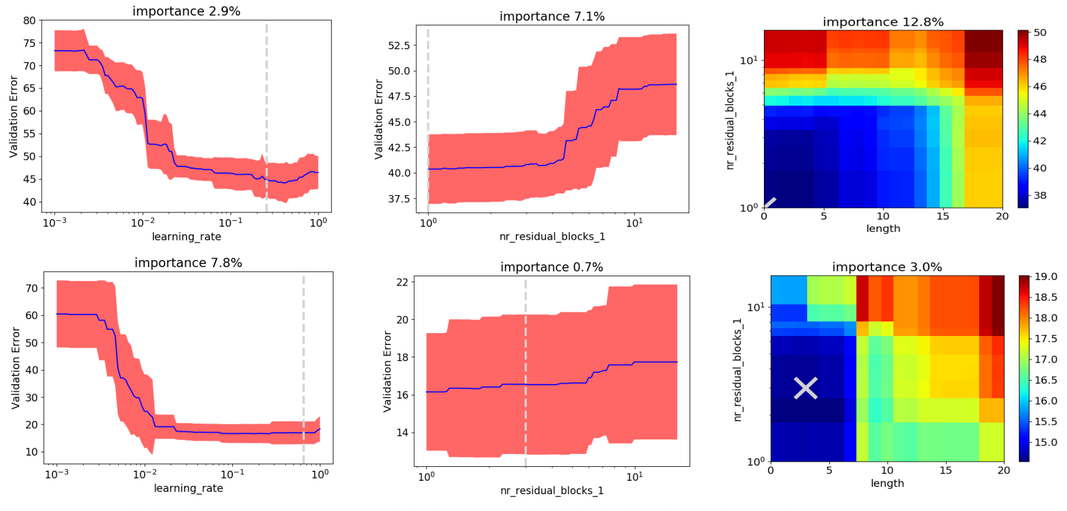
- **256 evaluations** on the full budget of 3h (32 GPU days)
- **Exploration-exploitation** trade-off:
  - explored sufficiently
  - covered good regions of the space
- Better results than manually constructed architectures that are part of the search space when trained for 3h
- **Optimizing jointly** architecture and hyperparameters beneficial

Analysis

- **Spearman rank correlation** between budgets performances
  - ➢ **Conclusion**: short runs ranking ⇏ long runs ranking

	1200s	1h	3h
400s	0.87	0.31	0.05
1200s		0.88	0.64
1h			0.86

- **fANOVA** – quantifies global importance of all parameters
  - ➢ **Conclusion**: strong interaction between architectural choices, hyperparameters and the training time



Top row: 400s budget Bottom row: 1h budget Gray dashed line/cross: best performance