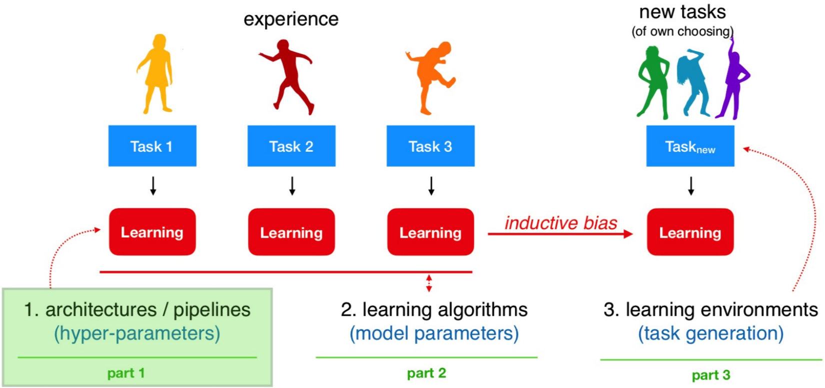
AutoML: Meta-Learning

Learning hyperparameter priors

Bernd Bischl Frank Hutter Lars Kotthoff
Marius Lindauer Joaquin Vanschoren

What can we learn to learn?

3 pillars



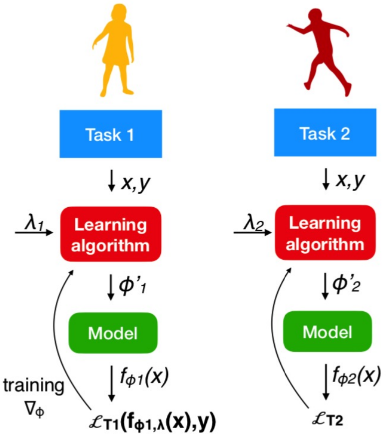
Terminology

Task: *distribution of samples* $q(x)$
outputs y*, loss* $\mathcal{L}(x,y)$

Learner: ***model parameters*** $\boldsymbol{\phi}$,
hyper-parameters $\boldsymbol{\lambda}$
optimizer

$$\phi^* = \mathrm{argmax}_\phi \log p(\phi \mid T)$$
$$= \mathrm{argmin}_\phi \mathcal{L}(f_{\phi,\lambda}(x),y)$$

Model: $f_\phi(x) = y'$



for reinforcement learning:
$q(x_1)$ + transition $q(x_{t+1} \mid x_t, y_t)$
$\mathcal{L}$ = (neg) reward

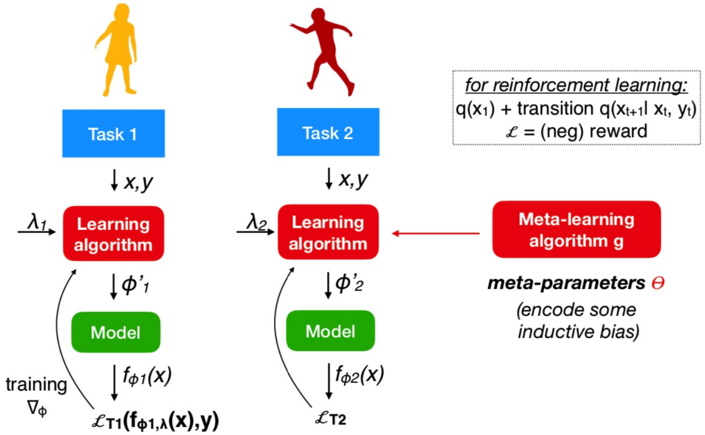
Terminology

Task: *distribution of samples $q(x)$*
outputs y, loss $\mathcal{L}(x,y)$

Learner: ***model parameters ϕ,***
hyper-parameters λ
optimizer

$$\phi^* = \mathrm{argmax}_\phi \log p(\phi \mid T)$$
$$= \mathrm{argmin}_\phi \mathcal{L}(f_{\phi,\lambda}(x),y)$$

Model: $f_\phi(x) = y'$

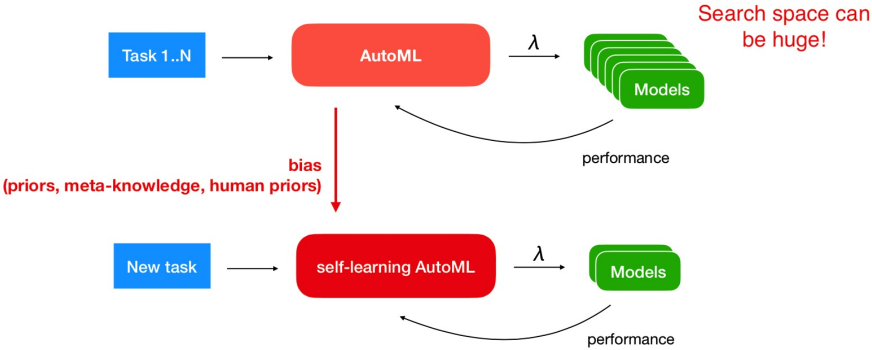


Learning hyperparameters

Closely related to Automated Machine Learning (AutoML)

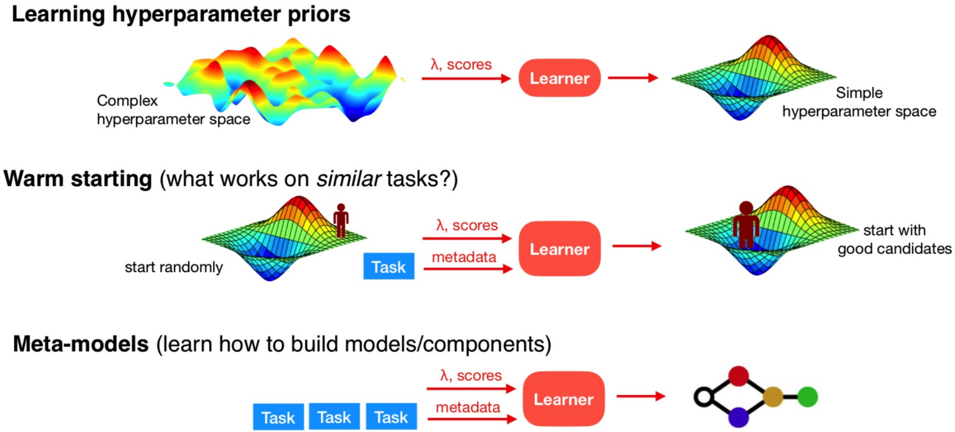
But:meta-learn how to design architectures/pipelines and tune hyperparameters

Human data scientists also learn from experience



Meta-learning for AutoML: how?

hyperparameters = architecture + hyperparameters

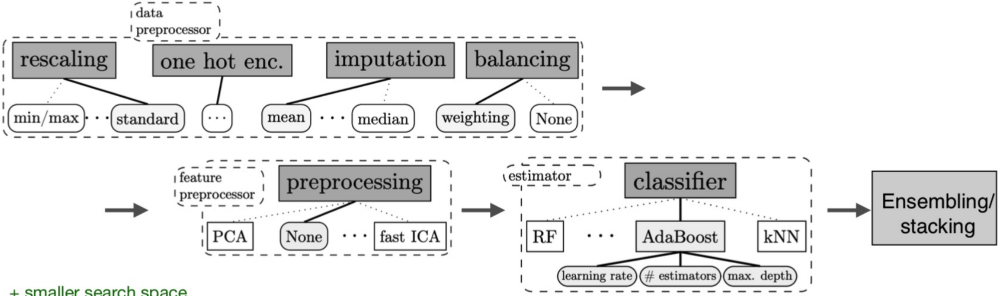


Observation:

current AutoML strongly depends on learned priors

Manual architecture priors

- Most successful pipelines have a similar structure
- Can we meta-learn a prior over successful structures?



+ smaller search space
- you can't learn entirely new architectures

Figure source: Feurer et al. 2015

Manual architecture priors

Successful deep networks often have repeated motifs (cells)

e.g. Inception v4:

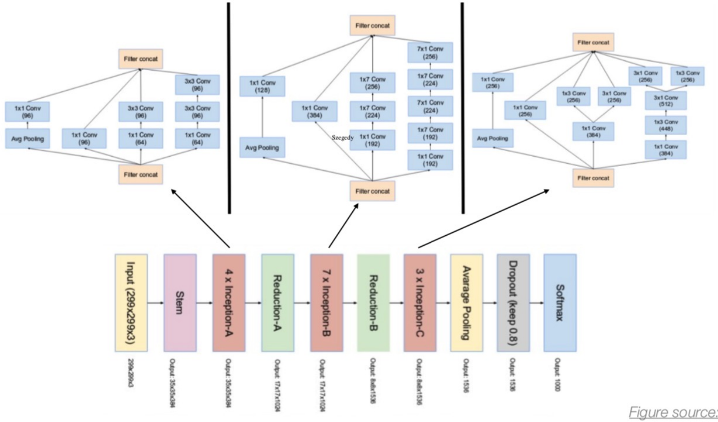


Figure source: Szegedy et al 2016

Cell search space prior

Compositionality: learn hierarchical building blocks to simplify the task

Cell search space

- learn parameterized building blocks *(cells)*
- stack cells together in macro-architecture

\+ smaller search space
\+ cells can be learned on a small dataset & transferred to a larger dataset
\- strong domain priors, doesn't generalize well

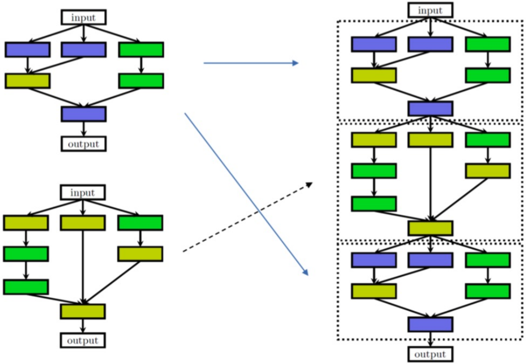


Figure source: Elsken et al., 2019

Cell search space prior

Compositionality: learn hierarchical building blocks to simplify the task

Cell search space

- learn parameterized building blocks *(cells)*
- stack cells together in macro-architecture

+ smaller search space
+ cells can be learned on a small dataset & transferred to a larger dataset
- strong domain priors, doesn't generalize well

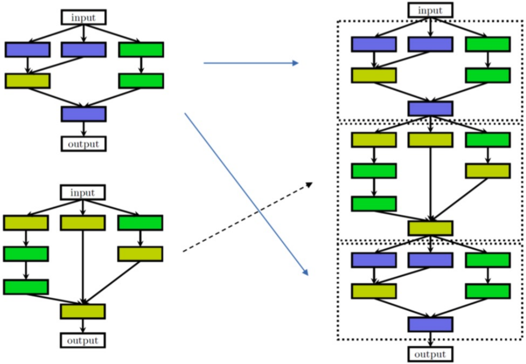


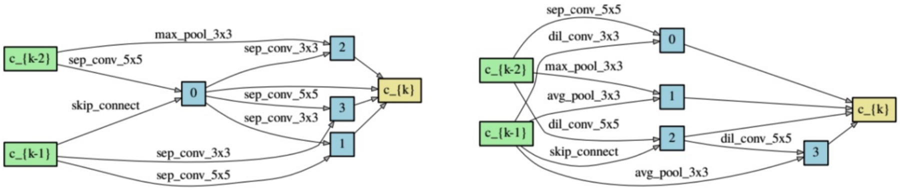
Figure source: Elsken et al., 2019

Can we meta-learn hierarchies / components that generalize better?

Cell search space prior

- Strong prior!

  If you constrain the search space enough, you can get SOTA results with random search!



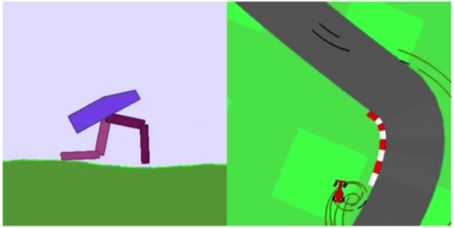
(a) Normal Cell (b) Reduction Cell

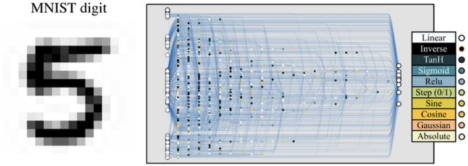
Convolutional Cells on CIFAR-10 Benchmark: Best architecture found by random search with weight-sharing.

Manual priors: Weight sharing

Weight-agnostic neural networks

- ALL weights are shared
- Only evolve the architecture?
  - Minimal description length
  - Baldwin effect?





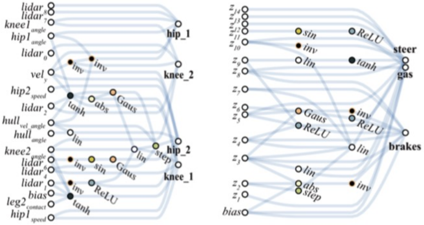
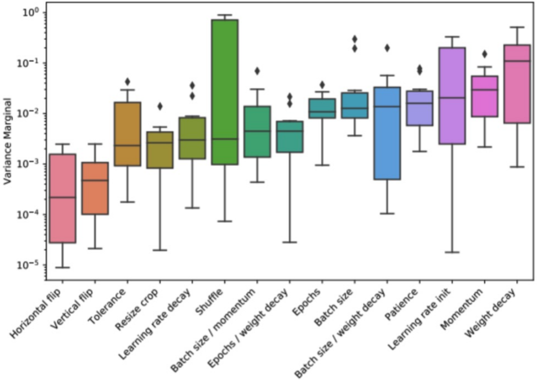


Figure source: Gaier & HA., 2019

Learning hyperparameter priors

Learn hyperparameter importance

- **Functional ANOVA** [1]
  - Select hyperparameters that cause variance in the evaluations.
  - Useful to speed up black-box optimization techniques



ResNets for image classification

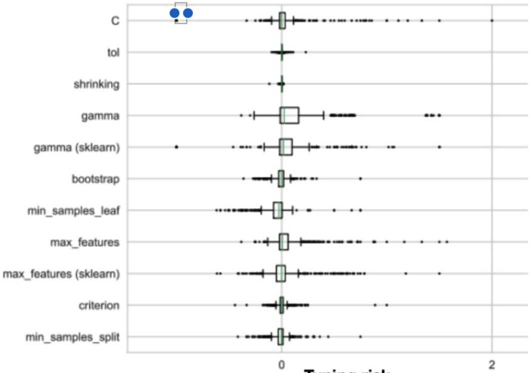
Figure source: van Rijn & Hutter, 2018

Learn defaults + hyperparameter importance

- **Tunability**
  Learn good defaults, measure importance as improvement via tuning

function
max_features
m = 0.16*p
m = p ^0.74
m = 1.15^sqrt(p)
m = sqrt(p)
gamma
m = 0.00574*p
m = 1/p
m = 0.006

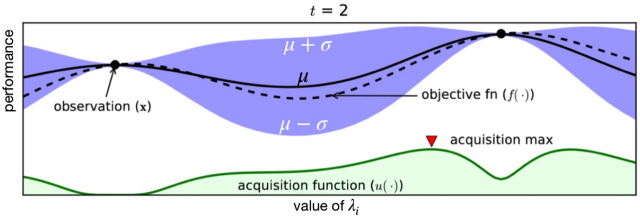
Learned defaults



Tuning risk

Bayesian Optimization (interlude)

- Start with a few (random) hyperparameter configurations
- Fit a surrogate model to predict other configurations
- Probabilistic regression: mean μ and standard deviation σ (blue band)
- Use an acquisition function to trade off exploration and exploitation, e.g. Expected Improvement (EI)
- Sample for the best configuration under that function



Bayesian Optimization

- Repeat until some stopping criterion:
  - Fixed budget
  - Convergence
  - EI threshold
- Theoretical guarantees
  - *Srinivas et al. 2010, Freitas et al. 2012, Kawaguchi et al. 2016*
- Also works for non-convex, noisy data
- Used in AlphaGo

Bayesian Optimization

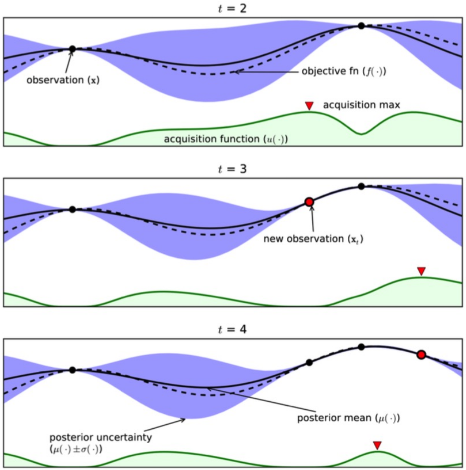
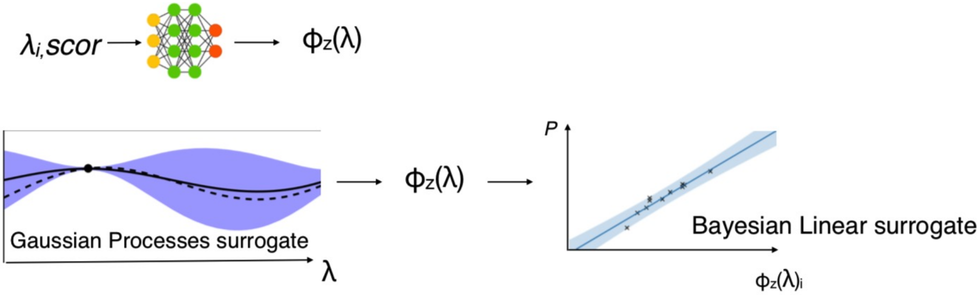


Figure source: Shahriari 2016

Learn basis expansions for hyperparameters

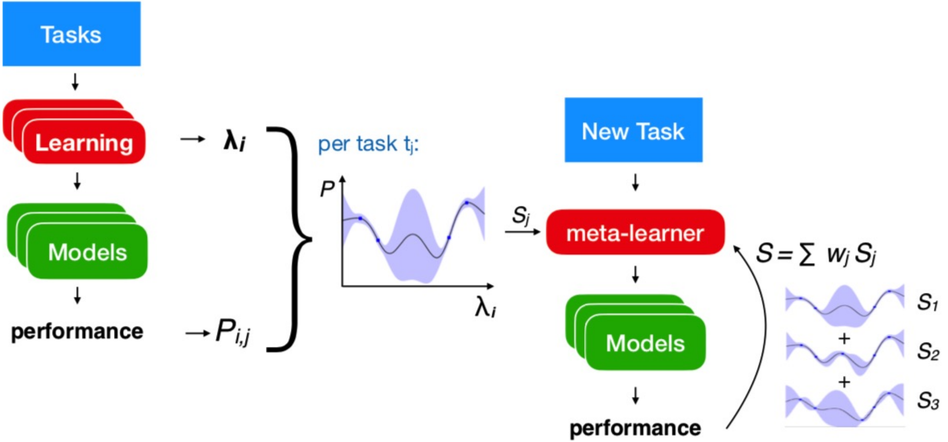
- Hyperparameters can interact in very non-linear ways
- Use a neural net to learn a suitable basis expansion $\phi_z(\lambda)$ for all tasks
- You can use Bayesian linear models, transfers info on configuration space

Learn basis expansion on lots of data (e.g. OpenML)



Surrogate model transfer

- If task j is similar to the new task, its surrogate model S_j will likely transfer well
- Sum up all S_j predictions, weighted by task similarity (as in active testing)
- Build combined Gaussian process, weighted by current performance on new task



Surrogate model transfer

- Store surrogate model S_ij for every pair of task i and algorithm j
- Simpler surrogates, better transfer
- Learn weighted ensemble $\rightarrow$ significant speed up in optimization