AutoML: Meta-Learning

Hyperparameter transfer from similar tasks

Bernd Bischl Frank Hutter Lars Kotthoff
Marius Lindauer Joaquin Vanschoren

Meta-learning for AutoML: how?

$$\text{hyperparameters} = \text{architecture} + \text{hyperparameters}$$

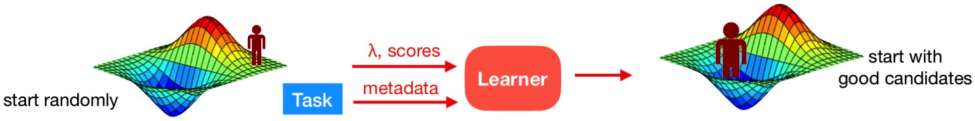
Learning hyperparameter priors

Warm starting (what works on *similar* tasks?)

Meta-models (learn how to build models/components)

Warm starting

(what works on similar tasks?)



How to measure task similarity?

- Hand-designed (statistical) meta-features that describe (tabular) datasets
- Task2Vec: task embedding for image data
- Optimal transport: similarity measure based on comparing probability distributions
- Metadata embedding based on textual dataset description
- Dataset2Vec: compares batches of datasets
- Distribution-based invariant deep networks

Warm-starting with kNN

- Find k most similar tasks, warm-start search with best λ_i
  - Auto-sklearn: Bayesian optimization (SMAC)
    - ★ Meta-learning yield better models, faster
    - ★ Winner of AutoML Challenges

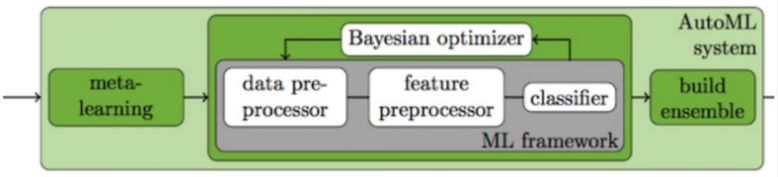
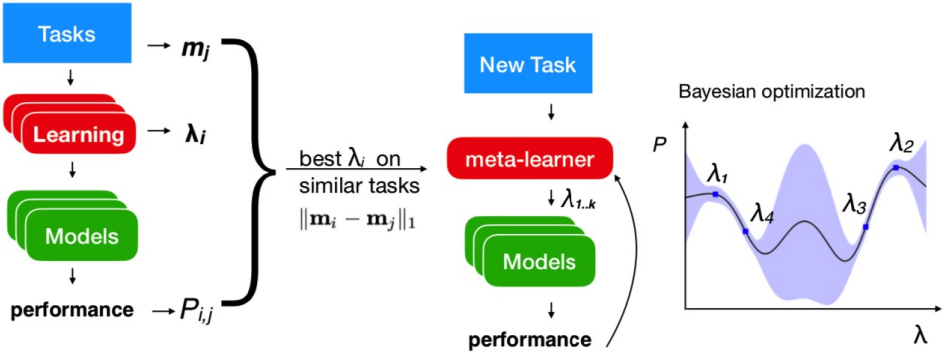


Figure source: Feurer et al., 2015

Warm-starting with kNN



Probabilistic Matrix Factorization

- Collaborative filtering: configurations λ_i are 'rated' by tasks t_j
- Learn latent representation for tasks T and configurations λ
- Use meta-features to warm-start on new task
- Returns probabilistic predictions for Bayesian optitmization

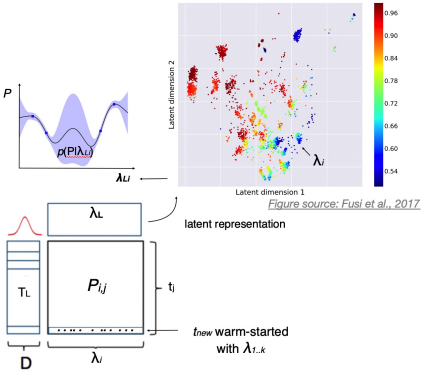
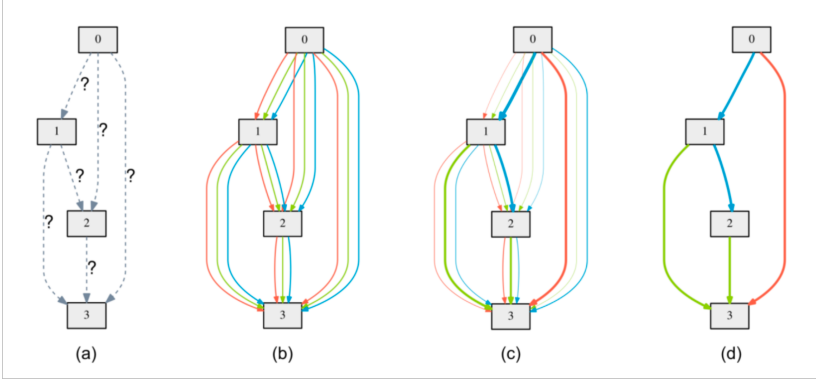


Figure source: Fusi et al., 2017

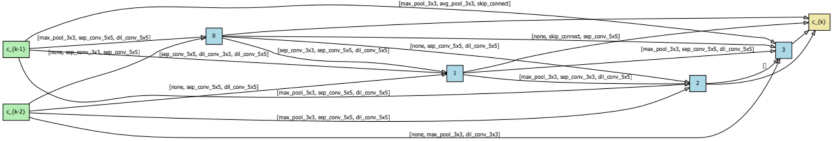
DARTS: Differentiable NAS

- Fixed (one-shot) structure, learn which operators to use
- Give all operators a weight α_i
- Optimize α_i and model weights ω_j using bilevel optimization
  - approximate $\omega_j * (\alpha_i)$ adapting ω_j after every training step

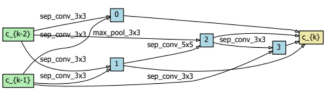


Warm-started DARTS

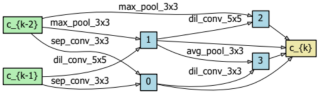
- Warm-start DARTS with architectures that worked well on similar problems
- Slightly better performance, but much faster (5x)



(a) FLOWER_V3 transfer architecture for normal cell



(b) Normal cell found on *aircraft* task



(c) Normal cell found on *birds* task

Meta-models

(learn how to build models/components)

Algorithm selection models

- *Learn* direct mapping between meta-features and $P_{i,j}$
  - Zero-shot meta-models: predict best λ_i given meta-features [1]

  - Ranking models: return ranking $\lambda_{1..k}$ [2]

  - Predict which algorithms / configurations to consider / tune [3]

  - Predict performance / runtime for given Θ_i and task [4]

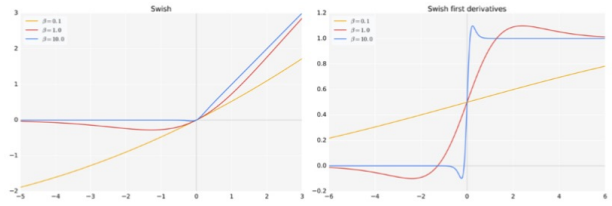
- Can be integrated in larger AutoML systems: warm start, guide search,…

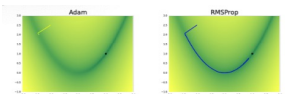
Learning model components

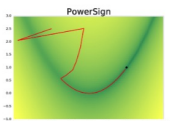
- Learn nonlinearities: RL-based search of space of likely useful activation functions
  - E.g. *Swish* can outperform ReLU

$$\text{Swish: } \frac{x}{1+e^{-\beta x}}$$

- Learn optimizers: RL-based search of space of likely useful update rules
  - E.g. PowerSign can outperform Adam, RMPprop
  - PowerSign : $e^{\text{sign}(g)\,\text{sign}(m)}g$ g: gradient, m:moving average
- Learn acquisition functions for Bayesian optimization

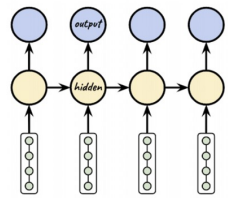


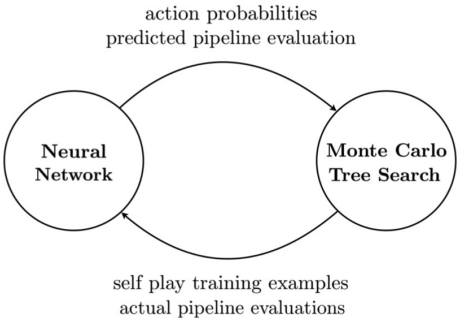




Monte Carlo Tree Search + reinforcement learning

- **Self-play:**
  - Game actions: insert, delete, replace components in a pipeline
  - Monte Carlo Tree Search builds pipelines given action probabilities
    - ★ With grammar to avoid invalid pipelines
  - Neural network (LSTM) Predicts pipeline performance (can be pre-trained on prior datasets)





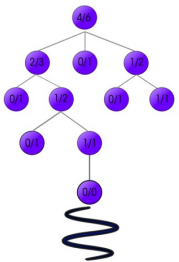
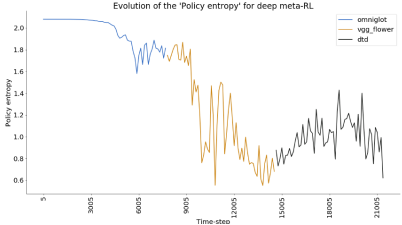


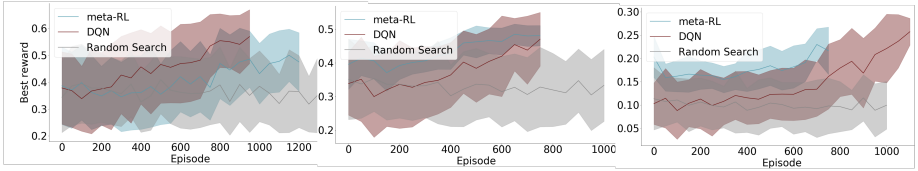
Figure source: Drori et al., 2019

Meta-Reinforcement Learning for NAS

Results on increasingly difficult tasks:

- Initially slower than DQN, but faster after a few tasks
- Policy entropy shows learning/relearning





vgg_flower dtd

MetaNAS: MAML + Neural Architecture Search

- Combines gradient based meta-learning (REPTILE) with NAS
- During meta-train, it optimizes the meta-architecture (DARTS weights) along with the meta-parameters (initial weights) θ
- During meta-test, the architecture can be adapted to the novel task through gradient descent

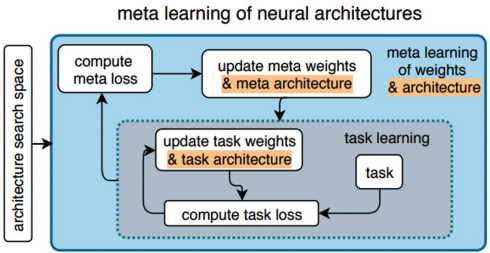


Figure source: Elsken et al., 2020