Hyperparameter Transfer Across Developer Adjustments

Danny Stoll¹, Jörg K.H. Franke¹, Diane Wagner¹, Simon Selg¹ & Frank Hutter¹,²

¹University of Freiburg, ²Bosch Center for Artificial Intelligence

In a Nutshell
- Developer of ML algorithms often perform adjustments like adding dropout.
- **Idea:** Meta learn across developer adjustments, not across tasks.
- **We** apply the across adjustments idea to hyperparameter optimization (HPO).

Baseline Algorithms
- $\mathcal{X}_{\text{std}}$: Standard HPO
- $\mathcal{X}_{\text{both}}$: HPO with transfer
- $\mathcal{X}_{\text{only-new}}$: HPO with only new steps

Experimental Setup
- We measure the speedup to reach a reference objective over TPE without transfer.
- We look at different budgets for the old and new HPO (10, 20, and 40 respectively).
- We measure the speedup across repetitions for each task, and then aggregate across tasks for each benchmark.

Type of Developer Adjustments
- **Add / Remove Hyperparameters**
- **Change Hyperparameter Range**
- **Change Algorithm / Hardware**

Benchmarks
- **FCN-A** Increase #units-per-layer 16x; Double #epochs; Fix batch size hyperparameter
- **FCN-B** Introduce per-layer choice of activation function; Change learning rate schedule from constant to cosine decay
- **NAS-A** Add 3x3 average pooling as choice of operation to each edge
- **NAS-B** Add node to cell template (adds 3 hyperparameters)
- **XGB-A** Expose four booster hyperparameters
- **XGB-B** Change four unexposed booster hyperparameter values
- **SVM-A** Change kernel; Remove hyperparameter for old kernel; Introduce hyperparameter for new kernel
- **SVM-B** Increase range for cost hyperparameter

Results
- Best first, transfer TPE, and their combination lead to large average speedups up to 1.2-2.9x.
- Only optimize new and drop unimportant never reach the objective in 20-70% of cases.