A Extra Experiments

A.1 Random Search and Bayesian Optimization Baselines

Here, optimize a model with $7,850$ and 10 hyperparameters, in which a separate L_2 weight decay is applied to the weights for each digit class in a linear regression model to assess hyper-trainings performance against other hyperparameter optimization algorithms. The conditional hyperparameter distribution and optimizer for the hypernetwork and hyperparameters is the same the prior experiments. Algorithm 3 is compared against random search and Bayesian optimization. Figure 7, right, shows that our method converges more quickly and to a better optimum than either alternative method, demonstrating that medium-sized hyperparameter optimization problems can be solved with Algorithm 3. Figure 7, left, shows that our method converges more quickly and to a better optimum than either alternative method, demonstrating that medium-sized hyperparameter optimization problems can be solved with Algorithm 3.

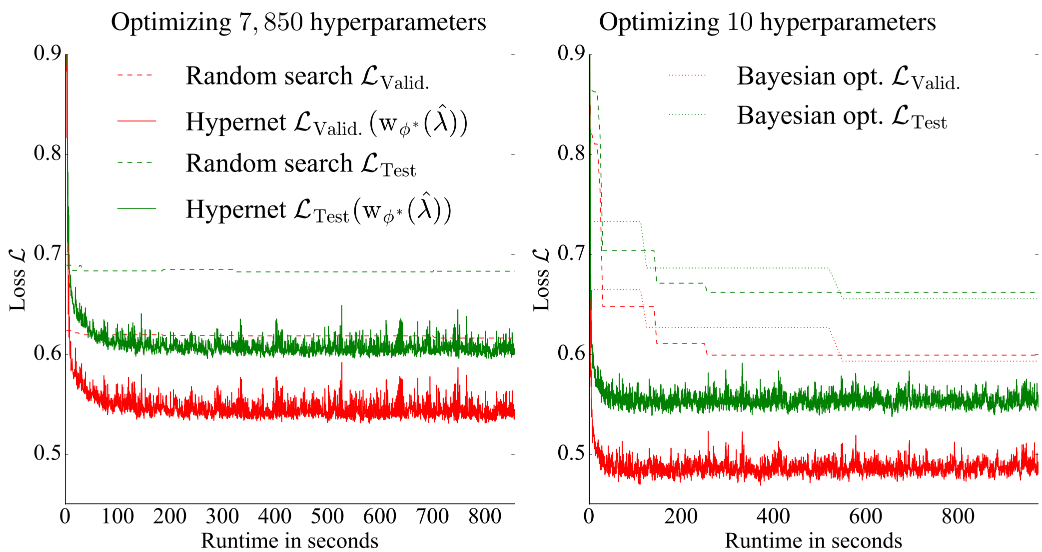

Figure 7: Validation and test losses during hyperparameter optimization. A separate L_2 weight decay is applied to the weights of each digit class, resulting in 10 hyperparameters. The weights w_{ϕ^*} are output by the hypernetwork for current hyperparameter $\hat{\lambda}$, while random losses are for the best result of a random search. hypernetwork-based optimization converges faster than random search or Bayesian optimization. We also observe significant overfitting of the hyperparameters on the validation set, which may be reduced by introducing hyperhyperparameters (parameters of the hyperparameter prior). The runtime includes the inner optimization for gradient-free approaches so that equal cumulative computational time is compared for each method.