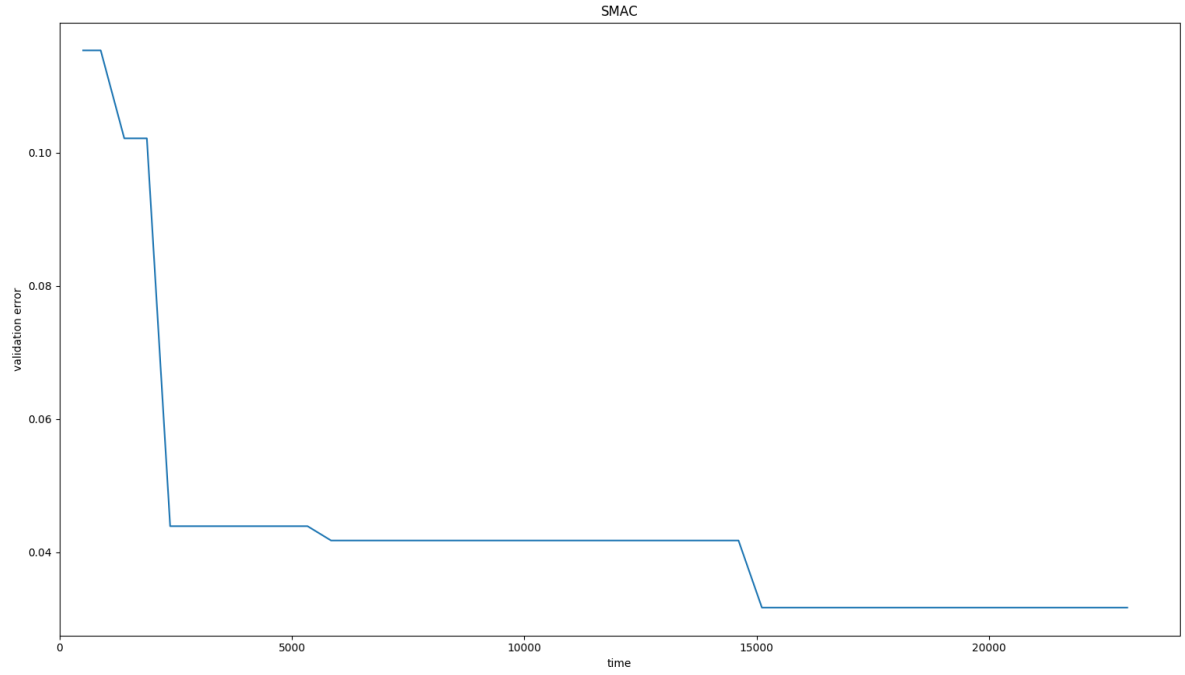
Deep Learning Lab
Exercise 6
Hyper-parameter Optimization - II

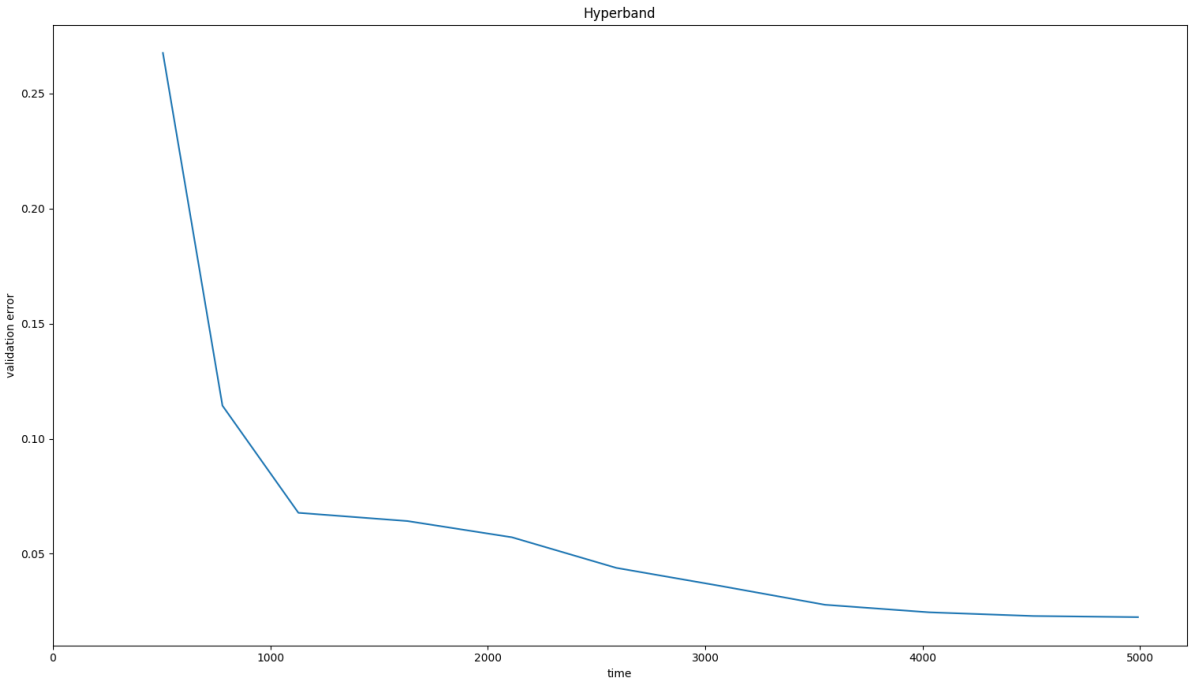
by
Muhammad Hamiz Ahmed

This exercise was a continuation of the previous exercise and was aimed to optimize hyper parameters in the configuration space, of a fully fully connected neural network for MNIST. The configuration space had to be created using python package called ConfigSpace. This time again, instead of using the true objective function, a surrogate function was used to save the development time.
Following configuration space was used:

Name	Range	Default	log scale	Type	conditioned on
Adam_final_lr_fraction	$[10^{-4}, 1.0]$	10^{-2}	✓	float	optimizer = Adam
Adam_initial_lr	$[10^{-4}, 10^{-2}]$	10^{-3}	✓	float	optimizer = Adam
SGD_final_lr_fraction	$[10^{-4}, 1.0]$	10^{-2}	✓	float	optimizer = SGD
SGD_initial_lr	$[10^{-3}, 0.5]$	10^{-1}	✓	float	optimizer = SGD
SGD_momentum	$[0.0, 0.99]$	0.9	-	float	optimizer = SGD
StepDecay_epochs_per_step	$[1, 128]$	16	✓	int	learning_rate_schedule = StepDecay
activation	{relu, tanh}	relu	-	cat	-
batch_size	$[8, 256]$	16	✓	int	-
dropout_0	$[0.0, 0.5]$	0.0	-	float	-
dropout_1	$[0.0, 0.5]$	0.0	-	float	num_layer ≥ 2
dropout_2	$[0.0, 0.5]$	0.0	-	float	num_layer ≥ 3
dropout_3	$[0.0, 0.5]$	0.0	-	float	num_layer = 4
l2_reg_0	$[10^{-6}, 10^{-2}]$	10^{-4}	✓	float	-
l2_reg_1	$[10^{-6}, 10^{-2}]$	10^{-4}	✓	float	num_layer ≥ 2
l2_reg_2	$[10^{-6}, 10^{-2}]$	10^{-4}	✓	float	num_layer ≥ 3
l2_reg_3	$[10^{-6}, 10^{-2}]$	10^{-4}	✓	float	num_layer = 4
learning_rate_schedule	{ExponentialDecay, StepDecay}	ExponentialDecay	-	cat	-
loss_function	{categorical_crossentropy}	categorical_crossentropy	-	cat	-
num_layers	$[1, 4]$	2	-	int	-
num_units_0	$[16, 256]$	32	✓	int	-
num_units_1	$[16, 256]$	32	✓	int	num_layer ≥ 2
num_units_2	$[16, 256]$	32	✓	int	num_layer ≥ 3
num_units_3	$[16, 256]$	32	✓	int	num_layer = 4
optimizer	{Adam, SGD }	Adam	-	cat	-
output_activation	{softmax}	softmax	-	cat	-

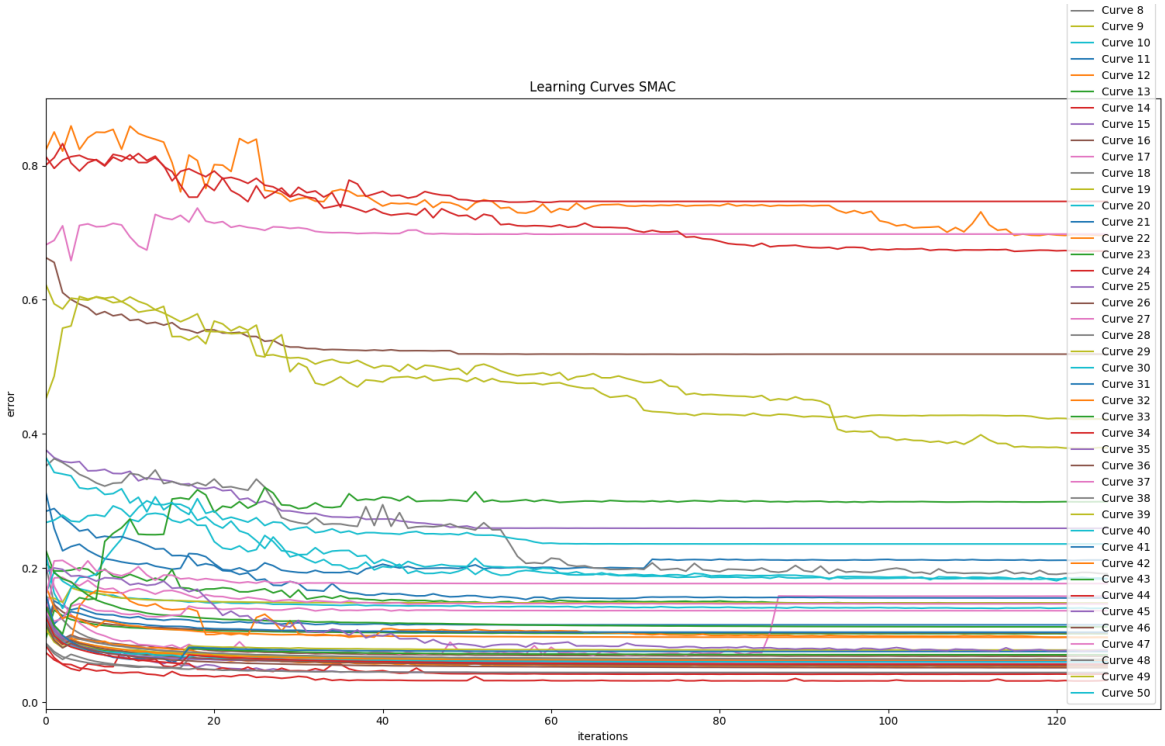
The hyper-parameters were optimized using SMAC and Hyperband. SMAC is an Bayesian optimization method that uses a random forest to model the objective function such that it is able to model mixed continuous, categorical and discrete configuration spaces. using Bayesian optimization and Random Search. The validation error plots of SMAC and Hyperband against wallclock time are as follows

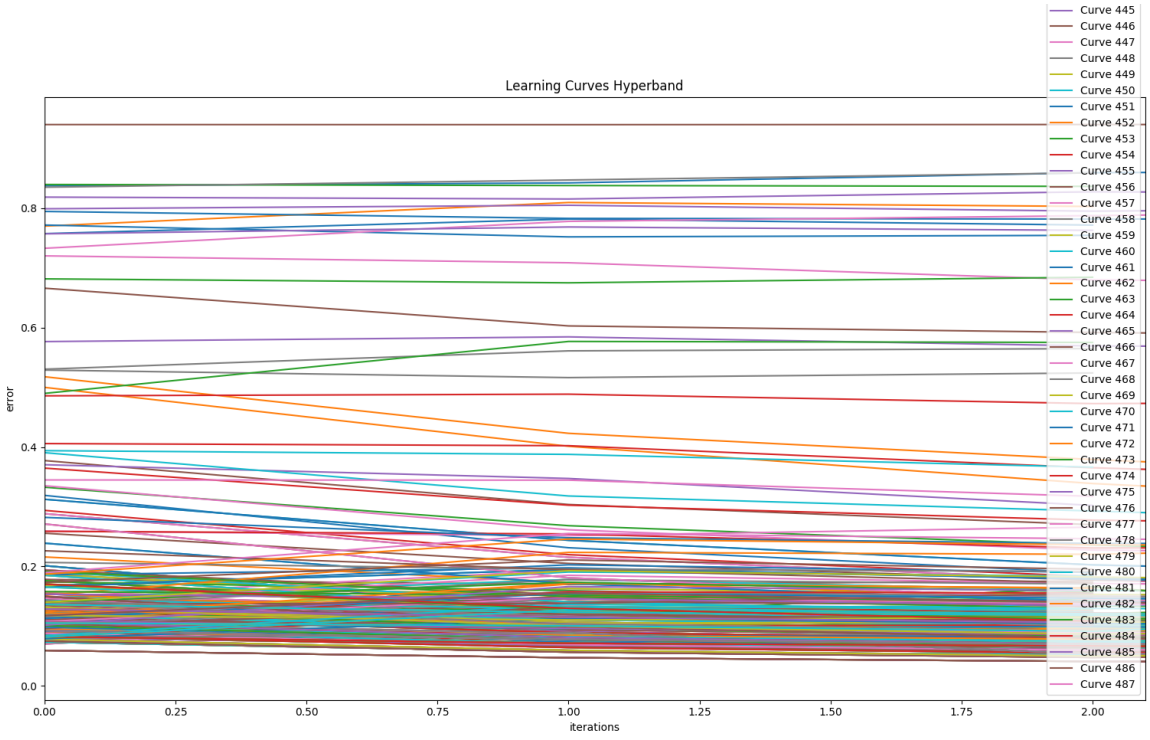




Hyperband takes less time than SMAC in reaching the lowest error value. This is because hyperband algorithm exploits the intuition that if a hyperparameter configuration is destined to be the best after a large number of iterations, it is more likely than not to perform in the top half of configurations after a small number of iterations.

Similarly, the learning curves of SMAC and Hyper-parameter are as follows:





SMAC has only 50 learning curves compared to hyperband which has around 487. In both the cases, the new curve starts from a lower error value than the previous one.