A Framework for Optimal and Efficient Neural Architecture Search

Christoforos Zisis

MSc. Machine Learning and Data Science
Supervisor: N.Giatrakos

Technical University of Crete, Spring 2024

Motivation

- Image Classification

- Finding the most suitable Machine Learning Algorithm for image classification.

- Finding the most suitable combination of Neural Networks for image classification.

- Finding the best combination of the above algorithms for image Classification in the shortest possible time.

Methodology

1. Benchmarker Software Module
2. Benchmark Statistics Collector and Benchmark Visualizer
3. Optimizer Module – BO Performance Modeler
4. Conclusions

Phase 1
Benchmarker Software Module

Phase 1: Benchmarker Software Module

Datasets

The software module uses the following datasets:

- CIFAR-10.
- CIFAR-100.

Parameters

The software module accepts the following parameters:

- Machine Learning algorithms.
- Neural Network combinations.

Description of Dataset

- The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class.
- The CIFAR-100 dataset is just like the CIFAR-10, except it has 100 classes containing 600 images each.

Figure: CIFAR-10 and CIFAR-100

Description of Parameters

Machine Learning Tecnhiques

- Support Vector Machines (SVM)
- K-Nearest Neighbors (KNN)
- Logistic Regression
- Decision Tree

NN Configuration Parameter Ranges

- Number of Convolution Layers: [2, 4, 6]
- Number of Dense Layers: [2, 4, 6]
- Number of Pooling Layers: [2, 4, 6]
- Number of Neurons Per Layer: [32, 64 ,128, 192, 256]
- Number of Batches: [4, 8 , 12, 16]
- Number of Epochs: [16, 32, 48, 64]

Phase 2
Benchmark Statistics Collector and Benchmark Visualizer

Phase 2: Benchmark Statistics Collector and Benchmark Visualizer

- A Statistic Collector extracts the **Accuracy** and the **Training Time** of the Benchmarker.
- The Scatter Plot will be produced with the Training time, Accuracy (y-axes) for each technique along with its Parameters (x-axes)

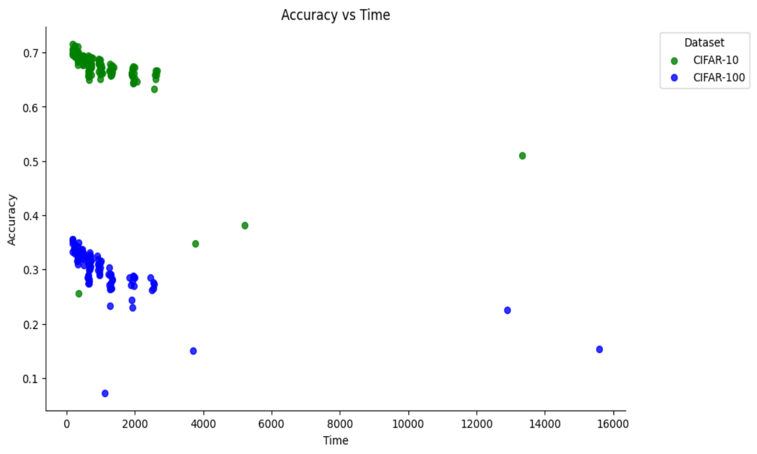
Benchmark Statistics Collector

Dataset	**Model***	**Accuracy**	**Time**
CIFAR-10	CNN_C4_D2_P4_N128_B16_E16	0.71600	171.33
CIFAR-10	CNN_C6_D2_P2_N64_B12_E16	0.71180	247.26
	⋮		
CIFAR-10	SVM	0.51024	13342.43

* [Convolution layers, Dense Layers, Pooling Layers, Neurons per layer, Batches, Epochs]

Scatter Plot

All Data Points (Cifar-10, cifar-100)



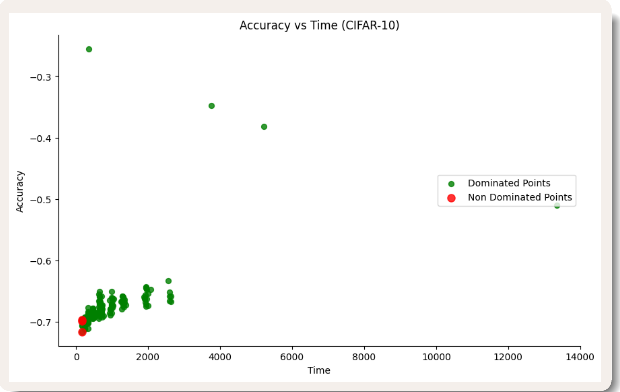
- How can the ideal points be identified, i.e., the points with the highest accuracy and the lowest training time?

Ideal Points For Cifar-10

- The ideal points are located towards the center of the axes.
- Multiplication of accuracy by -1.

Combinations/Models
[4, 2, 4, 128, 16, 16] *
[4, 2, 2, 192, 16, 16]
[4, 2, 2, 128, 16, 16]

Accuracy	Training Time
0.7160	171.339293
0.6991	170.925612
0.6970	170.419466

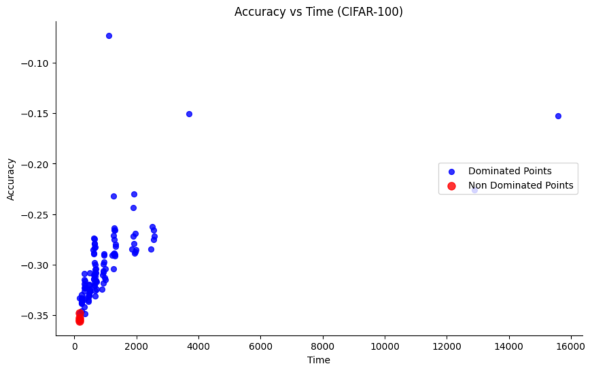


* [Convolution layers, Dense Layers, Pooling Layers, Neurons per layer, Batches, Epochs]

Ideal Points For Cifar-100

Combinations/Models
[4, 2, 4, 64, 16, 16]
[4, 2, 2, 192, 16, 16]
[4, 2, 2, 64, 16, 16]
[4, 2, 2, 128, 16, 16]

Accuracy	Training Time
0.3561	169.136173
0.3552	167.128322
0.3526	165.949351
0.3481	165.339114



Observation

- It is observed that for both CIFAR10 and CIFAR100, neural networks perform better than ML algorithms.

Phase 3
Optimizer Module – BO Performance Modeler

Phase 3: Optimizer Module – BO Performance Modeler

High Level Idea

In this phase the idea is that the Bayesian Optimizer developed, takes as input some initial points and gives the prediction for the other points (vectors).

- BO will be based on a surrogate model and an acquisition function.
- The surrogate model will be a Gaussian Process Regressor (GPR).

- Challenges in Developing a Bayesian Optimizer for Machine Learning Parameters
- The Bayesian Optimizer for the CNN's parameters.

Question

How accurately does the Bayesian optimizer predict the actual accuracy and training time?

Bayesian Optimizer for Machine Learning Algorithms

- The parameters of Machine Learning algorithms are not uniformly distributed.More specifically the search space include parameter configurations that are not applicable to all models.

Example

This vector is included in the search space, but no model exists with this combination of parameters:

- `['linear', 'ovo', '2', '1', 'l1', 'liblinear', 'auto', 'gini', 'best']`

Notation

Efforts were made to ensure high training times and low accuracy to avoid selecting these vectors. However, there are more such vectors than the available combinations.

`https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10302484&tag=1`

Bayesian Optimizer for Convolutional Neural Network

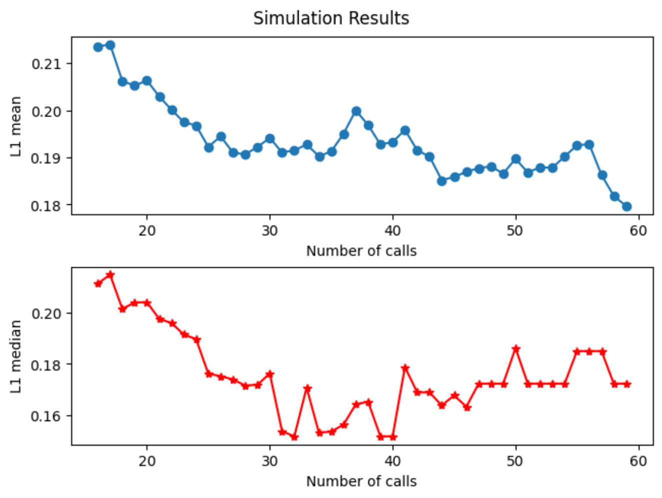
- The parameters of Convolutional Neural Networks are uniformly distributed.

Acquisition Functions: Lower Confidence Bound (LCB)

Initial Points: 16

Calls: 60

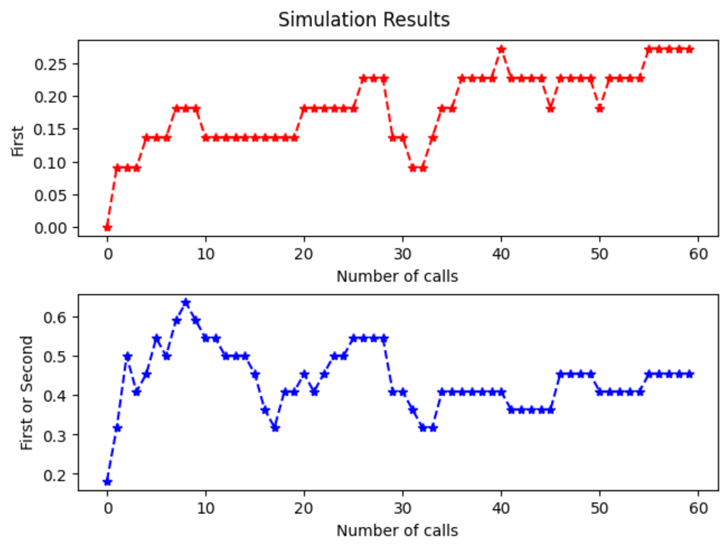
kernel: Rational Quadratic (length_scale=1.0, alpha=2.5)



Bayesian Optimizer for Convolutional Neural Network

First:
Measures how close we are to the best value over calls.

First Or Second:
Measures how close we are to first or second best value over calls.



Conclusions

Conclusions

Benchmarker Result

- Based on the real **Accuracy** and **Training Time**, the point (vector) $\mathbf{[4, 2, 2, 128, 16, 16]}$ * is one of the best combinations for both datasets.

Bayesian Optimizer Result

- Based on the **Bayesian Optimizer**, the point (vector) $\mathbf{[4, 2, 2, 128, 16, 16]}$ * is one of the best combinations.

- For CIFAR-100, Neural Networks with larger parameters are needed to achieve better accuracy.

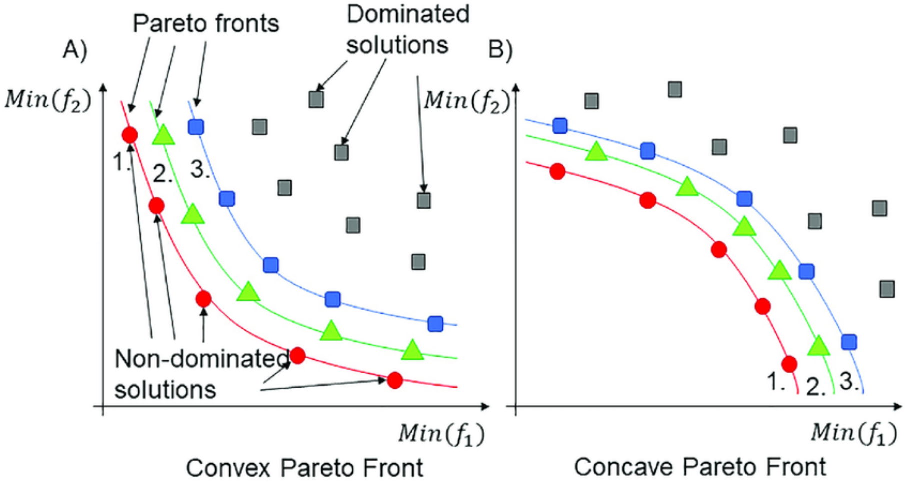
* [Convolution layers, Dense Layers, Pooling Layers,
Neurons per layer, Batches, Epochs]


That's all Folks!

References

- Orestis Plevris, Ali Nasir, *Journal Article from MDPI*, 2023, https://www.mdpi.com/2079-3197/11/7/147
- Errikos Streviniotis, Nikos Giatrakos, Yannis Kotidis, Thaleia Ntiniakou, Miguel Ponce de Leon, *Optimizing Resource Allocation for Tumor Simulations over HPC Infrastructures*, 2023, https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10302484
- Keras, *Deep Learning Framework*, 2023, http://www.keras.io/
- Alex Krizhevsky, *CIFAR-10 and CIFAR-100 datasets*, 2009, https://www.cs.toronto.edu/~kriz/cifar.html
- PyGMO, *Multi-Objective Optimization Utilities*, 2020, https://esa.github.io/pygmo2/mo_utils.html
- Scikit-learn, *Classification of Neighbors*, 2020, https://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html#sphx-glr-auto-examples-neighbors-plot-classification-py
- Scikit-learn, *Decision Trees*, 2020, https://scikit-learn.org/stable/modules/tree.html
- Scikit-learn, *K-means clustering on digits dataset*, 2020, https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_digits.html#run-the-benchmark
- Scikit-learn, *Logistic Regression*, 2020, https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
- Scikit-learn, *Support Vector Classification*, 2020, https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

Ideal Data Points



Machine Learning Techniques

- Automatic comparison of the following algorithms with their corresponding parameters:

Models	Parameters Ranges
SVM	$\left\{ \begin{array}{l} \text{Kernel} = \{\text{'linear'}, , \text{'rbf'}, \text{'sigmoid'}\} \\ \text{Decision Function Shape} = \{\text{'ovo'}, \text{'ovr'}\} \end{array} \right\}$
KNN	$\left\{ \begin{array}{l} \text{n\_neighbors} = [\text{start} = 2, \text{end} = 9, \text{step} = 2] \\ \text{p} = \{1, 2, \text{random()}\} \end{array} \right\}$
Decision Tree	$\left\{ \begin{array}{l} \text{Criterion} = \{\text{"gini"}, \text{"entropy"}, \text{"log\_loss"}\} \\ \text{splitter} = \{\text{"best"}, \text{"random"}\} \end{array} \right\}$
Logistic Regression	$\left\{ \begin{array}{l} \text{Penalty} = \{\text{'l1'}, \text{'l2'}, \text{'elasticnet'}, \text{None}\} \\ \text{Solver} = \{\text{'liblinear'}\} \\ \text{multi\_class} = \{\text{'auto'}, \text{'ovr'}\} \end{array} \right\}$