Sparkling Light Publisher

Sparklinglight Transtions on Artificial Intelligence and Quantum Computing(STAIQC)

Website: **https://sparklinglightpublisher.com/ ISSN (Online):2583-0732**

Hyperparameter Tuning of Deep learning Models in Keras

Mohamad Zaim Awang Pon[a], Krishna Prakash K K[b,*]

[a]CEO, AiSara, Jalan Tun Razak, Kaula Lampur, Malaysia
[b]Research Scholar, Srinivas University, Pandeshwar, Mangalore, Karnataka, India

Abstract

Hyperparameter tuning or optimization is one of the fundamental way to improve the performance of the machine learning models. Hyper parameter is a parameter passed during the learning process of the model to make corrections or adjustments to the learning process. To generalise diverse data patterns, the same machine learning model may require different constraints, weights, or learning rates. Hyperparameters are the term for these kind of measurements. These parameters have been trial-and-error tested to ensure that the model can solve the machine learning task optimally. This paper focus on the science of hyperparameter tuning using some tools with experimental values and results of each experiments. We have also documented 4 metrics to analyze the hyperparameter tuning results and benchmark the outcome.
The experimental results of two tools used commonly for deep learning models namely Keras tuner and AiSara tuner are captured in the article. All relevant experimental code is also available for readers in authors github repository. The metrics used to benchmark the results are accuracy, search time, cost and complexity and expalinability. The results indicate the overall performance of AiSara tuner in search time, cost and complexity and expalinability matrices are superior to keras tuners.

Keywords: Deep Learning, Model tuning, Hyperparameters, Hyperparameter tuning, Keras, Keras tuner, AiSara tuner, Machine learning

1. Introduction

Machine learning model tuning is usually a trial-and-error process through in which we change some hyperparameters, run the model on the data again with the objective of improving the performance to figure out which set of hyperparameters gives the best accurate model In my pursuit of building efficient deep learning models, the model tuning stage to get the best accuracy. In this paper accuracy-complexity balance depends on many hyperparameters and value of those parameters can have wide ranges. So, the challenge became clear. There is no single step that can give me one best solution. It involves several trials with hyperparameters and acceptable range of values. So we decided to master the science of tuning of deep learning models [1-4]. At first, we found a tuning approach using Keras Tuner. The example python script found in google colab was very useful. Soon we realized we have to go beyond Keras tuner as my goal was to design an efficient deep learning model. In this paper we found another tool called AiSara Tuner for tuning deep learning model.

E-mail address of authors: *[a]support@aisara.ai, [b,*]k rishna.prakash.kk@gmail.com*

Please cite this article as: Mohamad Zaim Awang Pon, & Krishna Prakash K K (2021). Hyperparameter Tuning of Deep learning Models in Keras. *Sparklinglight Transactions on Artificial Intelligence and Quantum Computing (STAIQC), 1*(1), 36–40. **ISSN (Online):2583-0732**

We have experimented with both the Keras tuners with various hyperparameters and benchmarked some measures to compare the tools. We have used fashion MNIST dataset and CIFAR-10 datasets for my experiments as it is convenient for anyone who wants to evaluate the tools from anywhere. The Table 1 shows different metrics used for benchmark. Fig.1 represents Images on Multiple layers of abstraction in deep learning [1].

Table1. Metrics used for benchmark

Sr. No.	Measure	Explanation
1	Accuracy (in %)	Highest accuracy obtained by the tuner with same set of parameters set by inputs
2	Search Time	Time taken by the tuner to get the search spaces and return the best parameters.
3	Cost and Complexity	The best set of parameters that minimizes the cost and maximises accuracy. Total number of learnable parameters will generally provide a pointer to computation cost in the training or prediction process. Fewer number of learnable parameters lower the computation cost.
4	Expalinability	How easy to explain the results from tuners.

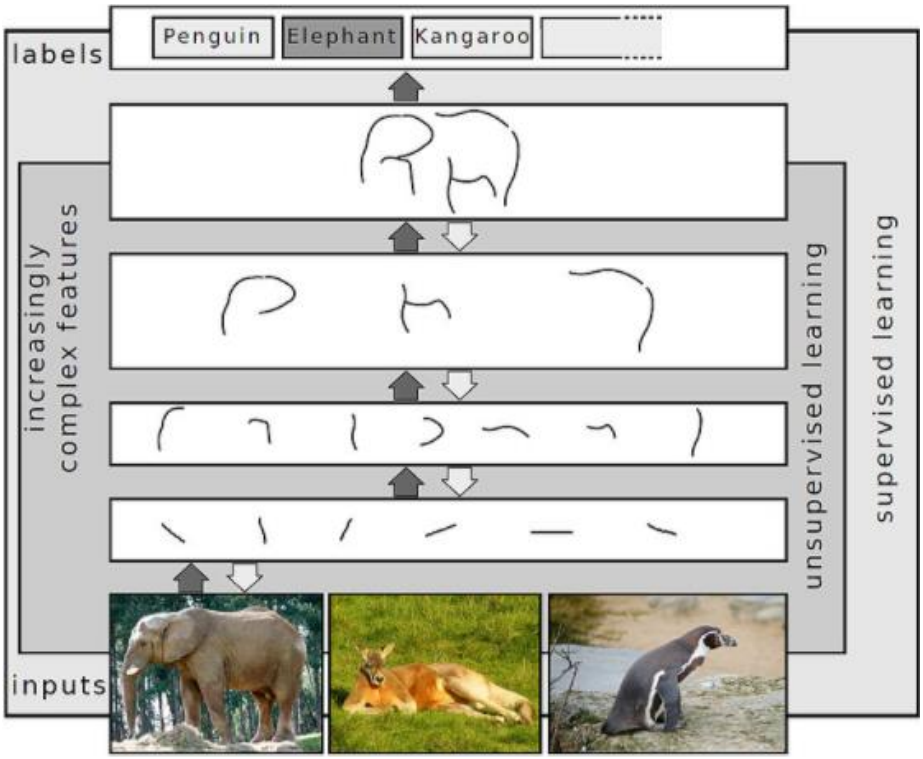

Fig. 1. Representing Images on Multiple Layers of Abstraction in Deep Learning [1]

2. Methodology and Experimental Setup

In this paper we have chosen a basic experimental dataset to enable all reviewers and learners to easily understand and verify the results by using commonly available dataset [5-6]. So the scripts provided here can be executed from Google Colab by anyone from anywhere. All the experiments are run in Google Colab and shared in my GitHub repo here [7-8].

2.1 System Environment

We have set Google Colab with Runtime hardware accelearator = "GPU" (Menu command Navigator: Runtime Change runtime type) [9-10]

2.2 Dataset

fashion MNIST dataset

```
import tensorflow as tf
from tensorflow import keras
```

```
# load fashion_mnist dataset
(img_train, y_train), (img_test, y_test) = keras.datasets.fashion_mnist.load_data()
```

CIFAR-10 dataset

```
(img_train, y_train), (img_test, y_test) = keras.datasets.cifar10.load_data()
```

2.3 Tuners

Keras tuners
We have separately set up mirror environment to Keras tuner and AiSara tuner in separate python scripts.
Keras tuner
Source of the python installer package: https://pypi.org/project/keras-tuner/

```
!pip install -q -U keras-tuner
import kerastuner as kt
```

AiSara tuner
Source of the python installer package: https://pypi.org/project/aisaratuners/1.1/

```
!pip install aisaratuners
from aisaratuners.aisara_keras_tuner import Hp, HpOptimization
```

2.4 Hyperparameters

We have used 3 hyperparameters number of layers, number of units, learning rate in my experiments.

```
# define Hps:
hps = Hp()
hp_1 = hps.numrange(name='num_layers',min=2,max=10)
hp_2 = hps.numrange(name='nodes_dense',min=64,max=512)
hp_3 = hps.numrange(name='learning rate',min=0.0001,max=0.01, type='log')
```

3. Experimental Results

The Fig. 2 shows the summary of all experimental results. Anyone can refer to GitHub report to view the source and results obtained from test runs.

		DNN Model		CNN Model		Pre-trained Model		Overall Performance
Sl no	Measure	Keras-tuner	AiSara-tuner	Keras-tuner	AiSara-tuner	Keras-tuner	AiSara-tuner	
1	Accuracy (in %)	87.91	87.46	92.7	92	84.3	83.4	Both tuners can provide same level of accuracy
2	Search time	3m 40s	2m 18s	8m 39s	5m 46s	21m 46s	7m 33s	AiSara is faster by 2X
3	Cost and complexity	143,905	110,515	4,633,455	1,968,071	15,503,962	14,826,315	AiSara can produce best model with 20% fewer number learnable parameters
4	Explainability	Limited	Excellent	Limited	Excellent	Limited	Excellent	AiSara tuner has built in functionalilty to visualize search space and results grapahically as well as tabular format. **AiSara is a clear winner with excellent presentation of search space and results**.

Fig. 2. Experimental Results

AiSara Keras tuner has excellent graphical visualization methods which can explain the search space. Few samples of visualizations provided by AiSara keras tuner shown below in Fig.3.

Fig. 3: Graphical visualization of aisasartuner search space

Fig. 4 explains the tabulated results of aisaratuner search space with optimization results.

```
[11] # aisara tuner optimization results in dataframe
     print(demo.opti_results)
```

num_layers_conv	nodes_dense	lr	loss	acc	val_loss	val_acc	Round	model ID
3	467	0.001000	0.086302	0.969083	0.253831	0.9256	Round_1	model_1_1
6	198	0.000158	0.135081	0.951233	0.213033	0.9249	Round_1	model_1_2
9	109	0.002512	2.303002	0.097367	2.302709	0.1000	Round_1	model_1_3
4	378	0.000398	0.082916	0.969783	0.250761	0.9256	Round_1	model_1_4
8	288	0.006310	2.303475	0.098967	2.302868	0.1000	Round_1	model_1_5
2	481	0.000398	0.078862	0.971117	0.248421	0.9292	Round_2	model_2_1
5	430	0.000338	0.085207	0.968667	0.229261	0.9319	Round_2	model_2_2
3	378	0.000367	0.086249	0.967483	0.234544	0.9248	Round_2	model_2_3
6	326	0.000432	0.092117	0.965967	0.229248	0.9268	Round_2	model_2_4
4	275	0.000468	0.088487	0.967317	0.230943	0.9317	Round_2	model_2_5
5	432	0.000329	0.081729	0.969667	0.230290	0.9273	Round_3	model_3_1
6	426	0.000338	0.090452	0.967183	0.210800	0.9338	Round_3	model_3_2
5	430	0.000334	0.085069	0.968950	0.283692	0.9207	Round_3	model_3_3
5	428	0.000348	0.087272	0.968350	0.225598	0.9306	Round_3	model_3_4
4	434	0.000343	0.084674	0.969433	0.233769	0.9312	Round_3	model_3_5

Fig. 4. Tabulated results of aisaratuner search space with optimization results

4. Conclusion

In this paper, we have reviewed the performance of the two popular hyperparameter tuning method using Keras tuner and AiSara tuner. We have used four metrics to benchmark the performance of the tuners. The results indicate the overall performance of AiSara tuner in search time, cost and complexity and expalinability matrices are better than keras tuners. This experiment will help all data science enthusiasts and deep learning buddies to understand the keras tuners.

References

[1] Schulz, H., & Behnke, S. (2012). Deep learning. *KI-Künstliche Intelligenz*, *26*(4), 357-363.
[2] Iman, R. L., Davenport, J. M., & Zeigler, D. K. (1980). *Latin hypercube sampling (program user's guide).[LHC, in FORTRAN]* (No. SAND-79-1473). Sandia Labs., Albuquerque, NM (USA).
[3] McKay, M. D., Beckman, R. J., & Conover, W. J. (2000). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, *42*(1), 55-61.
[4] Iman, R. L., Helton, J. C., & Campbell, J. E. (1981). An approach to sensitivity analysis of computer models: Part I—Introduction, input variable selection and preliminary variable assessment. *Journal of quality technology*, *13*(3), 174-183.
[5] Tang, B. (1993). Orthogonal array-based Latin hypercubes. *Journal of the American statistical association*, *88*(424), 1392-1397.
[6] Owen, A. B. (1992). Orthogonal arrays for computer experiments, integration and visualization. *Statistica Sinica*, 439-452.
[7] Ye, K. Q. (1998). Orthogonal column Latin hypercubes and their application in computer experiments. *Journal of the American Statistical Association*, *93*(444), 1430-1439.
[8] Keras (2021). Keras. Simple. Flexible. Powerful. Downloadable from https://keras.io/api/. Accessed on 21/07/2021.
[9] Keras (2021. KerasTuner. Downloadable from https://keras.io/keras_tuner/. Accessed on 23/07/2021.
[10] aisaratuners 1.1(2020). leveraging aisara algorithm for effective hyperparameter tuning. Downloadable from https://pypi.org/project/aisaratuners/1.1/. Accessed on 25/07/2021.

****