Hyperparameter Determines the Best Learning Curve on Single, Multi-Layer and Deep Neural Network of Student Grade Prediction of Pokhara University Nepal

Yagyanath Rimal, Palvi Pandit, Saikat Gocchait, Shariq Aziz Butt, Ahmed J. Obaid

rimal.yagya@pu.edu.np, Saikat.gocchait@sidtm.edu.in, shariq2315@gmail.com, ahmedj.aljanaby@uokufa.edu.iq palvi.pandit@gmail.com

Abstract. This primary study tries to explore the best machine learning model of hyperparameter tuning using single, multiple, and deep neural networks for predicting engineering student grade of Pokhara University Nepal in the year 2019. Generally, hyperparameter algorithms are adjusted from the data sets automatically when the model was designed. This is not a good idea for all sort of algorithm, with less flexibility of user choices. Therefore this article ties to meet the research gap between automatic calculation vs external model hyperparameter calculation on the same data sets applying comparison. The single neuron predicts 92 percent accuracy of the student's A, B, C, D grade of university examination with correlation with their internal marks respectively. The five-neuron multilayer neural network produces 72 percent accuracy and deep neural network with drop-out layer (32, 64) examines, the most suitable hyperparameter determines to produce an accurate result of 50 percent accurate when 32 hidden levels were used over 64 hyperparameters produce quite less accuracy (44 percent) of their final score in constituent campus Pokhara. The main objective of this study is to examine the comparison between various neural networks for educational settings for optimization of model deployment accurately when categorical large data prediction of student grade. Therefore, this study compares three comparisons of the neural network model with the same data sets for best hypermeter prediction.

Keyword. Hyper-parameter, Sum of Squared, Multilayer Perception, Cross-Entropy

1. Introduction

The machine learning model requires many parameters for optimal design and testing of research data classifies data into training and test data sets. While designing the model the internal configuration, the external configuration of many parameters for the implementation of the appropriate model execution [1]. Ultimately, this produces many predictions values such as p1, p2, p3, etc., model values are to be again compared and finalized after statistics evaluation as best fitted to the data sets [2]. This process starts with an incorrect parameter to get a precise parameter called hyperparameter [3]. In many cases, the average statistics and standard deviation were initially calculated based on the data and then adapted with the model design [4]. In the artificial machine learning model, the weight of input variables produces various value parameters that are used for predicting another level [5]. Similarly, in the support vector model use, external parameters and regression coefficient are the model parameters are generated from data itself [6]. Therefore, hypermetric are values other than the data generated for the
implementation and execution of model deployment. In many cases, the external parameters were taken by the model designer [7]. Although model designer's judgment on hyperparameters before the model parameters externally plays a vital role in many cases [8]. The hyperparameter was established as a heuristic, i.e. the self-discovery of its generated value automatically from the model. Similarly, in the K-NN model, the k value is established before the model designed [9]. The big difference between the hyperparameter and model parameter is the way, it will be set when designing the model [10]. On many occasions, alpha learning has been established externally using an iterative called hypermeters. Similarly, in the neural network, the hidden layers and their units of each iteration and the choice of activation functions have been established externally [11]. Therefore, those external parameters determine the learning algorithm based on the original parameters, the input parameters have been configured based on the hypermeter. Furthermore, in the deep learning model, the first alpha value with hyperparameter was generated and produced to validate the iteration process again [12]. What ultimately determines the new alpha value with many repetitions during the conversation [13]. Therefore, the hyperparameters are different for the different cases calculated during the design of the machine learning model [14] [15]. The hyperparameter is usually calculated as a babysitter model or as a training of many models. When unique data sets have to be determined, the new learning cos curve is determined. Similarly, when the GPU is large enough to configure the parallel model that plots many curves after analyzing the output that has drastically highs and lows, the best value pairs will be selected [16]. According to Wikipedia [17] in machine learning model always requires the best hyperparameters optimal hypermeters in learning algorithm controls the learning process with the value of another learned parameter. Because which directly control the behavior to train algorithm significantly impact on the model performance of both trained and test sample database. Modern supervised machine learning algorithms involve hyper parameters that must be set before executing them [18]. So the core of machine learning is adapting a model to the data after producing it with test sets [19]. The process of training a model with existing data to adapt to the model parameters is called model training [20] on the importance of hyperparameter adaptation, which parameters are important to adapt and which could be set to a value predetermined [21]. Which hyperparameters must be adjusted optimally for solving the machine learning problem. The adaptation process is of various types, such as manual search, random search, grid search, and Bayesian search [22].

The manual search uses the problem of guessing parameters and finding parameters at run time. Similarly, Grid Search is used to identify ranges for hyperparameters. Then, the evenly distributed parameters selected with the best performance alternative will be selected. Random search is used to identify ranges for hyperparameters [23] [24]. With multiple repetitions, until you find the parameters that work well, it should be compared and tends to work better than the other methods [25]. The Bayesian optimization approach to improve these approaches using the information obtained from the obtained hyperparameters [26]. Therefore, hyperparameters are crucial as they control the overall behavior of a machine learning model [27]. To minimize errors, models must be regularly modified by adjusting these hyperparameters which strongly influence the whole learning mechanism [28]. Therefore, appropriate machine learning increases algorithms for accurate predictive models for both educators and data scientists [29] [30]. Similarly, when designing the neural network model, the researcher would have had to establish several external hypermeters of 110 data samples from the internal markup database of the University of Pokhara using the neural network model such as n = neuralnet (SGPA ~SLC+Plus+Physics+Mathematics+Chemistry+Bio+ Mathematics2+int1+int2+ int3 + int4 + int5 + int6, data = training, hidden = c (1), err.fct = "sse", linear. output = FALSE) When designing the SGPA dependent variable model, a categorical final score for students in the fall 2019 result. Independent variables such as SLC, Plus, Physics, Chemistry, and their internal brands are predictors of the dependent SGPA degree. The two training data sets and the sample data sets of 70% and 30% of the data are divided by the test sets. The hidden level is an external hyperparameter that has been established with single or multiple levels of each complete neural network. Similarly, the "sse" error values will be set as the sum of the error squared when the dependent variables are more categorical, otherwise if true or false we use the cross-entropy "ce" for the differential function. The linear output will be false for
printing purposes. Instead of designing a single layer, the researcher can use multiple levels as hidden input = intermediate c (7.2), and the output levels are hyperparameters plot is the student's grade of the student's grade.

The first network use single neuron to convert its SGPA value of student grade classification whereas second plots describe two hidden layers with 7 and two intermediate hidden layers describe their weights with external bias terms. The lifesign=full indicates how much will be printed and the rep=5 another hyperparameter means each 5 times repetition than plots will be evaluated for better analysis prediction. This model produces the output as thresh is 0.01 rep is 1/3 steps, 1000 min thresh is 0.140899062108462 of each repetition and select the best alternatives when it will converse fewer errors as 51835error 1.24707 time is 9.26 secs when step three 55117error 1.74167 times is 9.86 secs converse its more errors and longer times. Which could easily have implied the first repetition is best with minimum errors and time to complication plot (n, rep=1) for the accurate final model.

A neural network with five neurons indicates the best model of student grade. Whose misclassification could be selected with rep=1 produces.

\[
\begin{array}{cccc}
\text{pred1} & 0 & 1 & 2 & 3 \\
0 & 0 & 31 & 10 & 38 \\
2 & 7 & 0 & 0 \\
\end{array}
\]

This output implies that there was first 0 category A grade with 2 data, the B grade 1 with 31 students, the C grade with 2 categories and Fail grade 38 students have no miss classification but in altogether 9(7+2) miss classification which is more than 92 percent accurate of student grade prediction on training.
on a calculation using 1- sum (diag (tab1)) / sum (tab1) is 0.90 percent accuracy. Similarly, the test data also has with miss classification of A grade and B grade students has 2 misclassification marks prediction whose accuracy is scored 1-sum (diag ( tab2 ) ) / sum ( tab2) [1] 0.7272727 percent.

```
pred2  1  2   3
0  6  2 13
1  1  0   0
```

Similarly, the model uses various algorithms the default is rsrop+ resilient backpropagation with weight backtracking and the stepmax is default 10000 which converse before these times of iteration. Similarly, the student records data will be tested with multi-layer hypermeter tuning after loading Keras, tensor flow, and tfruns packages for a deep multilayer network model. The same data sets were again loaded using reading using read.csv.

```
> data <- read.csv("C:/Users/user/Desktop/publication/baruda/SSS.csv")
```

The str(data) command displays the summary of statists of the data frame of 110 records of 14 variables.

$ SLC: num 0.9032 0.9355 0.0323 0.6774 0.5806 ... 

..............

$ SGPA: int 1 4 4 3 4 2 2 3 4 4 ... 

The na values in data sets were omitted as data=na. omit(data) function. And tested as is.na(data) testing the database with na or not display positive output. The variables SLC Plus Physics Math Chemistry Bio Math2 int1 int2 int3 int4 int5 int6 SGPA as 1 FALSE….FALSE.

```
> data=as. matrix (data) conversion matrix format 
> dimnames (data) = NULL making variable name of v1 v2 after deleting original columns headers names. num [1:110, 1:14] 0.9032 0.9355 0.0323 0.6774 0.5806 .
```

The normalization process of 13 independent variables are required to reduced time and memory management of all data analysis machine learning process as data [,1:13] =normalize (data [, 1 : 13 ] ) normalized. However, the data in dependent variables SGPA needs as numeric data [,14] =as. numeric (data [,14])-1 for model design describes as num [1:110, 1:14] 0.3105 0.3766 0.0228 0.3284 0.2517.

The data participating 70, 30 percent, and stored in testing and test of dependent and independent sample for model design. The one-hot encoding for again converting categorical dependent variables, when the model uses multi-class classification neural network design. The target attribute of the vector contains each class value to a matrix converted. The trainlabels= to_categorical (traintarget ) and testlabels = to_categorical (testtarget ) converts their values 01,2,3 respectively. Whose output will display as numb [1:110, 1:14] 0.3105 0.3766 0.0228 0.3284 0.2517.

Test of Deep Learning with Single Hyperparamenter

```
> runs=tuning_run ("experiment1.R", flags=list (dense_units=c (32,64)))
```

The tuning run command on tfruns package supports to run external files on flags sets the dense layers of 32 and 64 units proceeds with deep learning largely supports Keras, tensor flow package for multi-layer perceptron of multi-class classification of student different grades. The compile and fit the model visualize the training history and predict target values based on test data and it performs good results after adding layers and hidden layers for the optimization parameters to accomplish better results. Proceed with tuning run? [Y/n]: Y.

Training run 1/2 (flags = list (32))

```
FLAGS=flags(flag_integer('dense_units',32)) #hidden layer 32
model =keras_model_sequential () %>%
    layer_dense (units=FLAGSdense_units, activation = 'relu',
    input_shape = c (13)) %>%
```

#13 independent variables
The response variable SGPA is 4 unit's layers of student grade construct model using keras_model.Sequential() function when numerical data of fully-connected layers. The activation with keras and neural networks use relu activation. This rectifier used in a hidden layer of softmax activation function for output values are in the range 0 and 1 between predicts probabilities.

```r
layer_dropout(rate=0.1) %>% # 0.1 for avoiding 10% overfitting
layer_dense (units=16, # another dense layer
activation = 'relu') %>%
layer_dropout(rate=0.1) %>% # 0.1
layer_dense (units=8, # another dense layer
activation = 'relu') %>%
layer_dropout(rate=0.1) %>% # 0.1
layer_dense (units=4, activation='softmax')
```

The response variable SGPA is 4 unit's layers of student grade construct model using keras_model.Sequential() function when numerical data of fully-connected layers. The activation with keras and neural networks use relu activation. This rectifier used in a hidden layer of softmax activation function for output values are in the range 0 and 1 between predicts probabilities.

```r
model %>% compile (loss='categorical_crossentropy', optimizer='adam', metrics='accuracy')
```

The compiler use adam and the categorical_crossentropy loss function produces the accuracy of training accuracy its argument. The optimizer and loss arguments are required for model deployment. Here researcher may use stochastic gradient descent adams and rmsprop algorithm for fine tune. The loss function depends for a regression problem or mean squared error selection.

```r
history=model %>%
fit (training , trainlabels , epochs=50, batch_size=32, validation_split=0.2)
```

The model fits using both training and test samples for a specified 50 number of epochs of the training dataset. An epoch is a single pass to entire training and testing sets for verification. When model run the output will display as train on 70 samples, validate on 1 samples. Which runs 32 neurons first in between 13 independent with SGPA dependent variable.

```r
Epoch 1/50
```

```
Epoch 50/50
```

```
70/70 [==] - 2s 26ms/sample - loss: 1.0286 - accuracy: 0.5286 - val_loss: 1.0764 - val_accuracy: 0.3889
```

Training run 1/1 (flags = list(32))

![Figure 4](image)
The loss and acc indicate the accurate performance of training data while the val_loss and val_acc are for the test or validation data whose output will iterate with 64 hidden produced output as.

![Graph showing loss and accuracy](image)

**Figure 5.**
After conversing both the model the best hypermeter values will be analyzed and concludes the 32 hidden model will produce the best than 64 neuron model.

```r
> head(runs)
Data frame: 2 x 23
  run_dir metric_loss metric_accuracy metric_val_loss metric_val_accuracy
1  runs/2020-04-20T05-46-01Z      1.0343          0.5714          1.0767              0.4444
2  runs/2020-04-20T05-45-15Z      1.0661          0.4857          1.1183              0.5000
> runs[,c(5,6)]
Data frame: 2 x 2
  metric_val_accuracy flag_dense_units
1              0.4444               64
2              0.5000               32
```

Deep learning model with five hyperparameters

The deep neural network with five intermediate hidden neurons is designed using dense_units with 32,16 with two 0.1 dropout layer having 32 batch_size. The keras_model_sequential model with relu activation of 13 independent variables with dependent SGPA categorical data of student marks. This model produces 32 iterations with 2 of each model of 32 and 64 middle layers.

```r
FLAGS=flags(flag_integer('dense_units1',32), flag_integer('dense_units2',16),
flag_numeric ('dropout1',0.1), flag_numeric ('dropout2',0.1), flag_integer ('batch_size',32))
model =keras_model_sequential () %>%
  layer_dense(units=FLAGS$dense_units1, activation = 'relu',  input_shape = c(13)) %>%
  layer_dropout(rate=FLAGS$dropout1) # 0.1 for avoiding 10 % overfitting
  layer_dense(units=FLAGS$dense_units2, activation = 'relu') %>%
  layer_dropout(rate=FLAGS$dropout2) # 0.1
  layer_dense(units=4, activation='softmax') # our response is 4 grades of students layers
```

The compilation process using categorical cross entropy with optimizer adam and accuracy compilation and produces conversion as below.
model %>% compile (loss='categorical_crossentropy', optimizer='adam', metrics='accuracy')

for more than two layers
The model will fit with training, data with trainlabels with 50 epochs with omitting 20%
overfitting data.

history=model %>%
fit(training, trainlabels, epochs=50, batch_size=FLAGS$batch_size, validation_split=0.2)

Figure 6.
The both above line graphs describe the input and output variable of 32 and 64 hidden layers
with five dropdowns intermediate layer of each. The first plot considers losses and the second
describes the accuracy of each iteration. The loss is gradually decreased and becomes constant
after25 iterations whereas the accuracy is more or less constant during starting to end considers
linearly. Best hypermeter evaluation describes the model with loss and accuracy using the head function
as Data frame: 6 x 27.

| run_dir          | metric_loss | metric_accuracy | metric_val_loss | metric_val_accuracy |
|------------------|-------------|-----------------|-----------------|---------------------|
| 1 runs/2020-04-20T06-17-44Z | 1.0802      | 0.5000          | 1.0921          | 0.3889              |
| 3 runs/2020-04-20T06-16-33Z  | 1.0481      | 0.4857          | 1.0767          | 0.5556              |
| 10 runs/2020-04-20T06-14-49Z | 1.0554      | 0.4571          | 1.1052          | 0.4444              |

Figure 7.
> runs[,c(5,6)] Data frame: 32 x 2
metric_val_accuracy flag_dense_units1
1           0.3889                64
...........
3           0.5556                64
6           0.4444                32

> results=runs[,c(3,5:10)]
> results Data frame: 32 x 7
metric_accuracy  metric_val_accuracy flag_dense_units1 flag_den se_units2 flag_dropout1 flag_dropout2 flag_batch_size
1           0.5000              0.3889                64                32           0.2           0.2              64
--------------
3           0.4857              0.5556                64                16           0.2           0.2              64
……………….
6           0.4571              0.4444                32                32           0.1           0.2              64
………………………
32 0.5143    0.3889      32              16     0.1         0.1              32

2. Conclusion
The accurate marks prediction of student academic performance is significant for providing better educational services in higher education of Pokhara university engineering students. This study has investigated the effectiveness of single, multiple, and deep neural network techniques of student’s performance of their final grade. This comparison reveals the hyperparameter could be easily optimized when the single-neuron network of 90 percent accuracy results whereas multiple layers with five neurons produce 72 percent accuracy results. The deep neural network hyperparameter tested using five dropout layers of 32 neurons produces better-matched results of 64 neurons. Moreover, the estimation of student’s grade performance enabling timely support largely for better grade prediction using the machine learning model. Automatic machine learning methods in the educational field opens up new horizons for student grade prediction timely coming from educational settings.

References
[1] D. Anderson, "Data Model Design & Best Practices – Part 2," 2017.
[2] S. I. P. S. K. S. Prasad, Linear and nonlinear modeling approaches for urban air quality prediction, 2012.
[3] Y. ,. C. L. N. lXia, A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring, 2017.
[4] K. Academy, "The idea of spread and standard deviation, " https://www.khanacademy.org/math/probability/data-distributions-a1/deviation, 2019.
[5] H.-N. H. 1. a. Q. H. D. Jeng-Fung Chen 1, "Predicting Student Academic Performance: A Comparison of Two Meta-Heuristic Algorithms Inspired by Cuckoo Birds for," 2014.
[6] F. Bulbul, "What is the Weight vector parameter in Support Vector Machine in machine learning?," 2014.
[7] T. Danka, "How to properly ship and deploy your machine learning model," 2019.
[8] S. K. D. E. T. B. C. S. Annie Louis, Deep Learning to Detect Redundant Method Comments, 2018.
[9] M. Behroozi, "How can we find the optimum K in K-Nearest Neighbor," 2014.
[10] S. C. Smithson, Neural networks designing neural networks: multi-objective hyper-parameter optimization, 2016.
[11] A. S. Walia, "Activation functions and it’s types-Which is better?," 2019.
[12] J. S. Bergstra, Algorithms for Hyper-Parameter Optimization, 2011.
[13] T. I. Barna, "Whenever i run my neural network I get different result," 2016.
[14] J. Brownlee, "What is the Difference Between a Parameter and a Hyperparameter?," 2017.
[15] R. Yagyanath, "Machine Learning Cluster Analysis for Large Categorical Data Using R Programming," 2019.
[16] W. Jia, K. A. Shaw and M. Martonosi, Stargazer: Automated regression-based GPU design space exploration, 2012.
[17] Wikipedia, "Hyperparameter optimization," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Hyperparameter_optimization.
[18] B. B. A.-L. B. Philipp Probst, "Tunability: Importance of Hyperparameters of Machine Learning Algorithms," 2018.
[19] N. Srivastava, Dropout: A Simple Way to Prevent Neural Networks from, 2014.
[20] D. S. Ananya, "Hyperparameter Optimization and Why is it important?," 2018.
[21] H. Weerts, "Importance of hyperparameter tuning," 2018.
[22] Y. B. James Bergstra, Random Search for Hyper-Parameter Optimization, 2012.
[23] Y. Rimal, "Neural Network Machine Learning Analysis for Noisy Data R Programming," 2019.
[24] S. R. Young, Optimizing deep learning hyper-parameters through an evolutionary algorithm, 2015.
[25] D. G. A. V. S. James C. Hayton, Factor Retention Decisions in Exploratory Factor Analysis: a Tutorial on Parallel Analysis, 1999.
[26] S. Biswas, "Importance of Choosing the Correct Hyper-parameters While Defining a Model," 2019.
[27] A. Lee, "Why you should do Feature Engineering first, Hyperparameter Tuning second as a Data Scientist," 2019.
[28] P. Science, "Hyper parameter tuning: An important technique in machine learning!," 2018.
[29] G. K. S. K. Maria Tsiakmaki, "Implementing AutoML in Educational Data Mining," 2019.
[30] Y. Rimal, "Machine Learning Random Forest Cluster Analysis for Large Overfitting Data: using R Programming," 2020.