Deep learning algorithm for data classification with hyperparameter optimization method

T Badriyah, D B Santoso and I Syarif

Informatics and Computer Engineering Department, Electronic Engineering Polytechnic Institute of Surabaya (EEPIS), Politeknik Elektronika Negeri Surabaya (PENS), Keputih Sukolilo Surabaya 60111, Indonesia

Email: tessy@pens.ac.id

Abstract. The availability of data is an important thing in the decision-making process. With the trend of using Machine Learning and Data Mining, humans are facilitated in learning and finding hidden knowledge in the data. One technique used in the classification of data is the Deep Learning method. Deep Learning is widely used because of its ability to solve problems with satisfying results. In other words, Deep Learning has shown to be the one of the most powerful and widely used methods. However, the key of success depends on the setting the right parameters. With this reason, we need a way to optimize the use of hyperparameter in Deep Learning. The method commonly used in this case is Grid Search. However, this method requires a long time and high computation. While other methods such as Random Search and Bayesian Optimization, both are very reliable and efficient compared to Grid Search. The results of the experiments conducted in this study indicate that Bayesian Optimization is better than Random Search in performing hyperparameter optimization processes in Deep Learning.

Keywords—deep Learning; hyperparameter; grid search; random search; bayesian optimization

1. Introduction

Data can be analyzed and used as consideration for the decision making. It can be carried out with a variety of approaches such as using the Deep Learning method which is increasingly being used today because it is proven to be powerful in solving various problems. The forerunner of Deep Learning itself began in 1980 when Kunihiko Fukushima made Neocognition, the first model of the Convolutional Neural Network before being refined by Yann LeCun, Leon Bottou, Joshua Bengio and Patrick Haffner [1]. The basis of Deep Learning itself is Artificial Neural Network. Utilizing large computing power with data warehouse from images, videos, image text is very large [2]. Like other Neural Networks, in general Deep Learning architecture consists of visible and hidden layers where the weight of each perceptron unit is optimized using a backpropagation algorithm [3]. Examples of algorithms that apply Deep Learning include Deep Convolutional Neural Networks (DCNN) for image classification, Deep Belief Networks - Deep Neural Networks (DBN - DNN) for speech recognition, Recurrent Neural Networks (RNN) for language translation, Query-Oriented Deep Extraction (QODE) which is based on the Restricted Boltzmann Machine (RBM) for multi-document summarization, Conditional Restricted Boltzmann Machine (RBM) for Drug-Target Interaction (DTI) prediction and Deep Belief Network (DBN) for time-series data prediction [4].
Deep Learning is a representation-learning method which consists of a set of algorithms that are modeled with many levels of representation. Deep Learning can be done in a series of Deep Learning processes even when the computation and costs required are high, because it is necessary to determine the right parameters so that it can produce results expected.

The successful implementation of Deep Learning strongly depends on the hyperparameter settings used. There is no reference to determine the value of each hyperparameter that will be used so that it can produce good results. Therefore, a separate approach is needed to optimize each hyperparameter. In this study we propose the Deep Learning approach to classify data and perform hyperparameter optimization in Deep Learning, called Random Search and Bayesian Optimization. The use of these two methods aims to analyze the performance of both in hyperparameter optimization in Deep Learning and shows that the two methods are better than the methods commonly used in hyperparameter optimization (Grid Search) which requires large and long computations.

2. Related Works
Chen-Ying Hung et.all [5] compared DNN (Deep Neural Network) with three other approaches to predict stroke for 5 (five) years. The results show that DNN and the gradient boosting decision tree (GBDT) have the same accuracy as Logistic Regression and Support Vector Machine. DNN can be optimal using fewer patient data than GBDT. Deep Learning is a powerful machine learning approach [6]. However, Deep Learning performance is very dependent on the adjustment of hyperparameter so that a hyperparameter optimization process is needed. In general, the hyperparameter optimization method commonly used is Grid Search. However, grid search will take a very long time [5] [7] [8]. James Bergstra and Yoshua Bengio [9] made another approach, namely Random Search, the result with a smaller computing load Random Search resulted in the same even better modeling. The use of Manual Search and Grid Search requires computing and a long time even though the method is the easiest and most widely used. Then Jasper Snoek and Hugo Larochelle [10] conducted a Bayesian Optimization approach to search for hyperparameters, the result is Bayesian Optimization is better and better in finding the right hyperparameter in the CIFAR-10 case.

3. Methodology
This section will discuss the design and implementation of Hyperparameter Optimization on Deep Learning Algorithm for classification, including: system design, dataset used, data preprocessing, deep learning architecture, hyperparameter optimization and performance measurement.

3.1. System Design
Block diagram of system design is shown in Figure 1.

The system design developed starts from preprocessing data which includes the process of deleting data that is not needed because of it is not important (useless) such as id, name, other attributes that have no relevance to determining the target attribute value. Then handling missing values and continued with the process of standardization or normalization on the data. And the last stage of preprocessing is the use of feature selection methods by choosing a subset of features that are relevant and important in determining the value of target attributes.

Next section will be continued with an explanation of the dataset used in this study.
3.2. **Dataset Used**

The data in this study were obtained from the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/index.php) from UCI Repository, there are 3 (three) dataset that will be used in this study, namely: Arcene, Wine dan Wisconsin Diagnostic Breast Cancer (WDBC).

| No. | Dataset                             | Number of Data | Number of Attributes | Number of Class |
|-----|------------------------------------|----------------|----------------------|-----------------|
| 1   | Arcene                             | 10000          | 10000                | 2               |
| 2   | Wine                               | 178            | 13                   | 3               |
| 3   | Wisconsin Diagnostic Breast Cancer | 30             | 569                  | 2               |

3.3. **Data Preprocessing**

The purpose of preprocessing is to improve the quality of data before the data mining process is carried out. The preprocessing stages carried out in this study were Remove unimportant attributes, handling missing values, and Data Normalization / Standardization.
3.3.1. Remove unimportant attributes
At this stage the removal of attributes that are not relevant to the determination of target attributes such as id or name will be carried out. In addition, deletion is also carried out on empty data records or features that have variance equal zero if left unchecked will reduce the quality of the data itself.

3.3.2. Handling missing values
Missing values are caused by errors in the data input process or data extraction process. To handle missing values is used to fill an empty value using the average method, where the steps are as described in the following algorithm.

**Algorithm 1 : Fill Missing Value**

**Input :** Attribute contains missing value
**Output :** All Attribute value completed

For each value in Attribute
    If value == ‘NaN’
        value ← mean(Attribute contains missing value)
    End
End

3.3.3. Data Normalization
Normalization is used to improve data quality. We used z-score method which changes the data so that each attribute has a mean value of 1 (one) and standard deviation equal 1 (one). The following formula is the normalization of z-score:

\[
\text{new data} = \frac{x - \mu}{\sigma} \tag{1}
\]

In which \(\mu\) is the mean value and \(\sigma\) is the standard deviation value of the feature column.

\[
\mu = \frac{1}{N} \sum_{i=1}^{N} x_i \tag{2}
\]

\[
\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)^2} \tag{3}
\]

In which:
\(N\) = Number of data
\(x\) = data item

3.3.4. Feature Selection
Feature selection is the process of selecting a subset of relevant and important features that correlate with class output, so that the model produces better accuracy. In this study, genetic algorithms based on wrapper methods are used, which are some combinations of feature subset evaluated and compared with each other. The genetic algorithm itself is an algorithm inspired by evolution and natural selection. This method offers a good approach to finding solutions that are close to optimal for the optimization problem [11].
3.4. Deep Learning Architecture
At this stage hyperparameter optimization process is carried out, previously the hyperparameter that will be optimized first is defined by its values. There are many hyperparameters that are owned by Deep Learning, in this study there are seven hyperparameters that will be optimized, called hidden layer, hidden node, epoch, learning rate, activation function, batch size and dropout rate.

This study used Deep Learning algorithm with Multi-layer Perceptron (MLP) architecture or Fully Connected Layer or Deep Neural Network as shown in Figure 2.

![Deep Learning Architecture](image)

**Figure 2.** Deep Learning architecture.

Inputs to this process are features in each dataset to be tested. The features have been selected in the feature selection process. The output of the classification process is the label class of each dataset.

The following is the process of Deep Learning with the Multi-layer Perceptron (MLP) architecture:

1. Initialize the weight randomly and also initialize the topology and other hyperparameters.
2. Then the feedforward process which passing passing neurons in the hidden layer and calculating the output and target
3. If both are not the same, then update weight process or the backpropagation process. This process is done until it finds a weight with an error approaching 0 (zero) or until the iteration is complete.

In training there is a dropout process that is removing or ignoring neurons or nodes in the hidden layer, this process can also prevent overfitting. The Adam method is used to determine the weight value, this method is easy to implement and computationally efficient [12].

3.5. Hyperparameter Optimization
At this stage hyperparameter optimization process is carried out, previously the hyperparameter that will be optimized is defined by its values. There are many hyperparameters that are owned by Deep Learning. In this study there are seven hyperparameters that will be optimized.

This study uses the Random Search method and Bayesian Optimization as part of hyperparameter optimization. The following will explain each method.

3.5.1. Random Search
The Random Search method works almost the same as Grid Search, but the Random Search will not evaluate the whole model, but only evaluates as many as iterations or samples from the models that are formed.

**Algorithm 2 : Random Search**

```
Input : IterationNum, HyperparameterDict
Output : BestModel
For (Iter $\in$ IterationNum)
    DeepLearningAlgorithm(Model) ← SelectRandom(HyperparameterDict)
    If Score(Model) < Score(BestModel)
        BestModel ← Model
End
Return(BestModel)
```
3.5.2. Bayesian Optimization
Bayesian Optimization uses a gaussian process which is a function distribution. With this process can be able to calculate the Expected Increase from each point in the search space. Bayesian Optimization usually gives a non-trivial value, off-the-grid which sometimes takes values outside those specified but are still in the hyperparameter dictionary range, for example if a batch size 32, 64, 128 can be initialized to produce a value of 115 from the results of the Bayesian Optimization process.

Algorithm 3 : Bayesian Optimization

\[
\text{Input : } \text{IterationNum, HyperparameterDict} \\
\text{Output : } \text{BestModel} \\
\text{Set : } f \leftarrow \text{Gaussian Process(HyperparameterDict)} \\
\text{For } (\text{Iter} \leq \text{IterationNum}) \\
\quad \text{DeepLearningAlgorithm(Model)} \leftarrow \text{Select}(f) \\
\quad \text{If } \text{Score(Model)} < \text{Score(BestModel)} \\
\qquad \text{BestModel} \leftarrow \text{Model} \\
\text{End} \\
\text{Return (BestModel)}
\]

3.6. Performance Measurement
To evaluate the performance of the algorithm, the confusion matrix values are True Positive (TP), True Negative (TN), False Positive (FP), Negative Phase (FN), for example in class X:

| actual values | prediction values |
|---------------|-------------------|
| TP            | TRUE              |
| FP            | FALSE             |
| FN            | TRUE              |
| TN            | FALSE             |

Table 2. Precision, recall and accuracy.

Accuracy is a metric score used to calculate how many correct prediction results from the classification generated by the algorithm. Accuracy can be formulated as follows:

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4} 
\]

Precision is a metric score used to calculate precision or data with class A on the results of data classified as class A. Precision can be formulated as follows:

\[
\text{Precision} = \frac{TP}{TP + FP} \tag{5} 
\]

Recall is a metric score used to calculate the amount of data correctly predicted as class A classifications divided by all data that have actual classification values as class A. Recall can be formulated as follows:

\[
\text{Recall} = \frac{TP}{TP + FN} \tag{6} 
\]

F1-measure is a metric score that calculates the balance value of precision and recall, F1-measure can be formulated as follows:
\[ F1 = 2 * \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \]  
\[ \alpha + \beta = \chi \]  

4. Experiment and Analysis

In this section, the experiment was carried out with cross validation or 10-fold and the metric score measured using accuracy, precision, recall and f1-measure.

The experiment test has been done first without going through the hyperparameter optimization process, and then it is done with the hyperparameter optimization results in each dataset used.

4.1. Dataset Arcene

In Arcene dataset there is an increase in scoring results from before optimization and after optimization. The hyperparameter results generated by the Random Search are not so significant. Moreover the scoring generated by Bayesian Optimization optimization it is not even good, so that in this dataset both methods cannot improve the scoring results.

Table 3. Comparison of classification scoring before and after optimization on arcene dataset.

| Performance Measurement | Before Optimization | Random Search | Bayesian Optimization |
|-------------------------|---------------------|--------------|-----------------------|
| Accuracy                | 0.51                | 0.58         | 0.77                  |
| Precision               | 0.52                | 0.57         | 0.80                  |
| Recall                  | 0.50                | 0.58         | 0.76                  |
| F1                      | 0.49                | 0.53         | 0.75                  |

4.2. Dataset Wine

In the Wine dataset, there was a significant increase in scoring results from both Random Search and Bayesian Optimization results. The results from Bayesian Optimization are better than Random Search.

Table 4. Comparison of classification scoring before and after optimization on wine dataset.

| Performance Measurement | Before Optimization | Random Search | Bayesian Optimization |
|-------------------------|---------------------|--------------|-----------------------|
| Accuracy                | 0.58                | 0.85         | 0.94                  |
| Precision               | 0.57                | 0.85         | 0.94                  |
| Recall                  | 0.58                | 0.86         | 0.95                  |
| F1                      | 0.53                | 0.86         | 0.94                  |

4.3. Dataset WDBC

In the WDBC dataset, there was a significant increase in scoring results from both Random Search and Bayesian Optimization results. Both Random Search and Bayesian Optimization both provide improved scoring results.

Table 5. Comparison of classification scoring before and after optimization on wdbc dataset.

| Performance Measurement | Before Optimization | Random Search | Bayesian Optimization |
|-------------------------|---------------------|--------------|-----------------------|
| Accuracy                | 0.56                | 0.95         | 0.95                  |
| Precision               | 0.33                | 0.95         | 0.95                  |
| Recall                  | 0.50                | 0.95         | 0.95                  |
| F1                      | 0.36                | 0.93         | 0.95                  |
From all experiments it can be concluded that Bayesian Optimization is superior to Random Search, in the term of accuracy. However, the optimization results in these two methods cannot find the best hyperparameter, it requires more iterations so that the sampling to be evaluated in the Deep Learning method becomes more and more extensive search.

5. Conclusion
This research optimizes Deep Learning algorithm by using hyperparameter optimization method. The dataset used was obtained from the UCI Machine Learning Repository called Arcene, Wine, and the Wisconsin Diagnostic Breast Cancer (WDBC). The system design developed including feature selection is a genetic algorithm that can improve the accuracy of the Deep Learning method.

To provide optimal results, Deep Learning requires the proper setting of hyperparameter. While the optimization of hyperparameter in Deep Learning used in this study uses Random Search and Bayesian Optimization. In the Arcene dataset both Random Search and Bayesian Optimization have not produced significant results. In the Wine dataset Bayesian Optimization was superior to Random Search. Furthermore, in WDBC datasets, both Bayesian Optimization and Random Search methods produce equally good performance improvements.

In the future works, this study will explore Deep Learning architecture that can enable better results. In addition, it is also necessary to try other methods to optimize hyperparameters in Deep Learning to obtain better performance results.

References
[1] Barakbah A R 2017 Convolutional Neural Network & Deep Learning
[2] Parloff R “Why Deep Learning is Suddenly Changing Your Life Fortune”, 2016. [Online]. Available: http://fortune.com/ai-artificial-intelligence-deep-machine-learning/. [Accessed 2018]
[3] LeCun Y, Bengio Y, Hinton G 2015 Nature (Toronto: Macmillan Publishers Limited) vol 521 pp 436-444
[4] Firmansyah and Gultom Y Aplikasi Deep Learning dalam Berbagai Domain : Review Paper.
[5] Hung C-Y, Chen W-C, Lai P-T, Lin C-H, Lee C-C 2017 39th Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society (EMBC) (Seogwipo: IEEE) vol 39 pp 3110-3113
[6] Assodiky H, Syarif I, Badriyah T 2017 Int. Electronics Symp. On Knowledge Creation and Intelligent Computing (IES-KCIC)
[7] Xiao T, Ren D, Lei S, Zhang J, Liu X 2014 Proceeding of the 11th World Congress on Intelligent Control and Automation (Shenyang: IEEE) vol 11 pp 1529-1533
[8] Qolomany B, Maabreh M, Al-Fuqaha A, Gupta A, Benhaddou D 2017 13th Int. Wireless Communications and Mobile Computing Conference (IWCMC) (Valencia: IEEE) vol 13 pp 1285-1290
[9] Bergstra J and Bengio Y 2012 Random Search for Hyper-Parameter Optimization Journal of Machine Learning Research 13(1) 281-305
[10] Snoek J, Larochelle H, Adams R P 2012 26th Annual Conf. on Neural Information Processing Systems (Lake Tahoe, NV: Elsevier) vol 4 pp 2951-2959
[11] Yang J and Honavar V 1998 Feature Subset Selection Using A Genetic Algorithm (Boston MA: Springer) pp 117-136
[12] Kingma D P and Ba J L 2015 3rd Int. Conf. for Learning Representations (San Diego: Cornell University Library)