공학석사학위논문

베이지안 기반 데이터 클래스 가중치 최적화를 통한 딥러닝 모터 결함 진단

Deep Learning Approach for Motor Diagnosis using Bayesian Based Class Weight Optimization

2021 년 2 월

서울대학교 대학원

기계항공공학부

신용진

베이지안 기반 데이터 클래스 가중치 최적화를 통한 딥러닝 모터 결함 진단

Deep Learning Approach for Motor Diagnosis using Bayesian Based Class Weight Optimization

지도교수 윤 병 동

이 논문을 공학석사 학위논문으로 제출함

2020 년 10 월

서울대학교 대학원

기계공학부

신 용 진

신 용 진의 공학석사 학위논문을 인준함

2020 년 12 월

위 원 장 : 김 도 년 (인)

부위원장 : 윤 병 동 (인)

위 원 : 신 용 대 (인)

Abstract

Deep Learning Approach for Motor Diagnosis using Bayesian Based Class Weight Optimization

Yongjin Shin
Department of Mechanical and Aerospace Engineering
The Graduate School
Seoul National University

Diagnosis of motor defects are essential task, because the defects can lead to failure of an entire system, causing deterioration in quality of applications and user dissatisfaction. Recently, this problem has been addressed by a data-driven approach based on deep learning methods. However, in real industrial environment, defect data are insufficient compared to the normal data, which significantly degrades the learning performance of the diagnostic model. This paper proposes a deep learning-based diagnosis method, defining weight balancing parameters to solve the class imbalance between normal and defect data. The parameters can make the model to more focus on the defect data during training. We optimized the parameters through Bayesian method, and find the best model to improve classification performance in minor classes. Experimental results show that the model with optimized parameters enhanced performance in given imbalanced data. This refers that the model can proceed training without editing the input data to balance between minor and major classes.

Keywords: Motor defect
Data imbalance
Weight balancing parameter
Loss function
Bayesian optimization

Student Number: 2019-22241

Table of Contents

Abstract i

List of Tables v

List of Figures vi

Chapter 1 Introduction 1

1.1 Motivation 1

1.2 Dissertation Layout 2

Chapter 2 Theoretical Backgrounds 4

2.1 Loss Function 4

2.2 Evaluation Index of Classification Performance 6

2.3 Bayesian Optimization 7

2.3.1 Surrogate Model 8

2.3.2 Acquisition Function 9

2.4 STFT(Short-time Fourier transform) 10

Chapter 3 Proposed Idea 12

3.1 Weight Balancing Parameters 12

3.2 Hyper-parameters Optimization using Bayesian Method 13

Chapter 4 A Experimental Process 16

4.1 Data Summary..16
4.2 Preprocessing..17
4.2.1 Sliding window filter..17
4.2.2 STFT 2D Image..18
4.3 Neural Network..19
4.4 Optimization Setting..21
4.5 Comparison Methods..22
4.5.1 Under Sampling..22
4.5.2 Over Sampling..23
4.5.3 Focal Loss..24

Chapter 5 Result & Analysis..25
5.1 Classification Performance..25
5.2 Feature Visualization..28
5.3 Application for Different Imbalance Level..30
5.4 Comparison with Other Hyper-parameter Optimization Methods....................31

Chapter 6 Conclusion..34

References 35

국문 초록 37

List of Tables

Table 4-1 Defect types of motor. 16

Table 4-2 Composition of training, validation, and test data. 20

Table 4-3 # of data sets for each classes using under sampling 23

Table 4-4 Shift size and # of windows for each classes using oversampling.... 24

Table 5-1 Classification metrics for each case. 26

List of Figures

Figure 2-1 Learning process of neural network. 5

Figure 2-2 Combination of observation and prediction. 6

Figure 2-3 Bayesian optimization using GP [5]. 9

Figure 3-1 Flow chart of learning model update using Bayesian optimization. 15

Figure 4-1 Raw data examples: (a) normal, (b) defect 1, (c) defect 2, (d) defect 3. 17

Figure 4-2 Sliding window filter. 18

Figure 4-3 STFT 2D image examples: (a) normal, (b) defect 1, (c) defect 2, (d) defect 3. 19

Figure 4-4 Neural architecture of 2D-CNN. 21

Figure 5-1 Normalized confusion matrix: (a) case ①, (b) case ②, (c) case ③, (d) case ④, (e) case ⑤, (f) case ⑥. 27

Figure 5-2 Feature visualization through t-SNE: (a) case ①, (b) case ②, (c) case ③, (d) case ④, (e) case ⑤, (f) case ⑥. 29

Figure 5-3 F1 score and loss for each cases: (a) Intermediate imbalance level, (b) low imbalance level, (c) high imbalance level. 31

Figure 5-4 Types of hyper-parameter optimization methods. 32

Figure 5-5 F1 score vs epochs for each optimization method: (a) Random search (b) Bayesian optimization. 33

Chapter 1

Introduction

1.1 Motivation

Motor defects cause unexpected motor failure during operation. If the motor failure occurs, the entire system is shut down and can lead to human accidents as well as economic losses at the production line Therefore, it is essential to diagnose motor defects in advance, and many studies have been conducted related to this. In particular, recently, many approaches using deep learning for motor diagnosis have appeared [1]. On the other hand, there are several big issues with the deep learning approach. One of the main issues is data imbalance between classes. The defect data in the actual field is not sufficient, so that makes diagnosis using deep learning difficult. In order to solve the problem, various methods have been tried from a data perspective and an algorithm perspective [2].

From the data point of view, the data imbalance problem is solved by balancing the input data itself by class. Oversampling is a method of adjusting the input by augmenting data of a minority class to match the level of the majority class, and under sampling is a method of adjusting the input by removing data of the majority classes according to the level of the minority class. In addition, under the data imbalance, Generative Adversarial Networks (GAN) is used as a method of generating data of the minority class [12]. From an algorithmic point of view, data imbalance can be resolved by transforming the loss function in deep learning. (Focal Loss, dice loss)

Dice loss and *GDL (generalized dice loss)* are introduced to resolve the data imbalance [3]. *Mean false error (MFE)* and *mean squared false error (MSFE)* are proposed to make up for shortcoming of mean squared error loss function [2]. *Focal loss* are used in convolutional neural networks to enhance classification performance of the minority class [9]. However, attempts to perform deep learning-based motor diagnosis by solving data imbalance from an algorithmic perspective are still insignificant.

This study aims to resolve the imbalance between normal and defective data by defining weight balancing parameters for each class in the loss function of a neural network. In addition, by setting the defined weight balancing parameter as a model hyper-parameter and performing Bayesian optimization, an optimized model is found and the classification performance is maximized within a given model [4].

1.2 Dissertation Layout

Including this section, this paper is organized with 6 sections. Section 2 provides background knowledge for the proposed idea. In section 3, weight balancing parameters and hyper-parameter optimization using Bayesian method are explained in detail. Section 4 shows a case study for the proposed idea. Results and analysis are carried out in Section 5 to verify the effectiveness of the suggestion method, where classification performance is discussed. Section 6 summarizes the whole research and gives future works.

Chapter 2

Theoretical Backgrounds

2.1 Loss Function

In deep learning, a neural network is composed of an input layer, hidden layers, and an output layer. Model parameters at each nodes are automatically determined by the model structure. When data enters the input layer in a specific shape, it is transmitted through calculations with internal parameters at each node of the hidden layers. As it reaches the output layer after the hidden layer, final predicted values of the model come out.

The loss function is a numerical indicator of how well the model learned the data. This function can be defined using the difference between the model's output value and the user's desired output value. After one epoch process of transferring data from the input layer to the output layer is finished, the parameters inside the model are updated in the direction of reducing the loss function through backpropagation. The process repeats again until the model has learned enough of the data. As a result, the model learns the data by repeating the process of updating the parameters inside the model in a direction that minimizes the defined loss function.

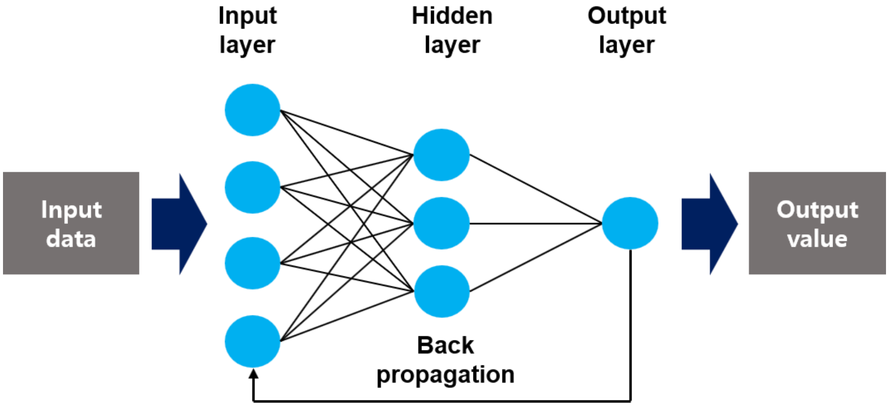


Figure 2-1 Learning process of neural network.

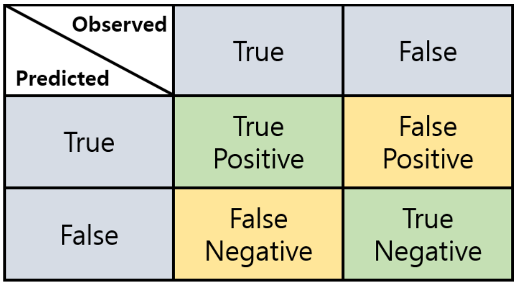
The loss function is expressed as Equation (2.1). Assuming the number of samples, real output value, and output value of the model are expressed as $n, y_i, and\ \hat{y}_i$, respectively. In general, the loss function is calculated as the average of the loss values of all data samples. Types of loss functions that are frequently used include MSE(Mean squared error), RMSE(Root mean squared error), binary cross-entropy, categorical cross-entropy and so on. In this study, we use categorical cross-entropy because we handle multiple classification problems. Categorical cross-entropy is calculated by Equation (2.2), where t_i is a ground truth; s_i is the i^{th} element of the last layer's output in score vector; C is the number of classes.

$$L(\hat{Y}) = \frac{1}{n}\sum_{i}^{n} L\,(y_i, \hat{y}_i) \tag{2.1}$$

$$CE = -\sum_{i}^{C} t_i \log(f(s)_i)$$
$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \tag{2.2}$$

2.2 Evaluation Index of Classification Performance

When there are multiple classes of data, the data can be classified into True and False depending on whether or not they are included in a specific reference class. Then, the multiple classification problem can be viewed as a double classification problem that matches whether or not it is a reference class. In the double classification problem, there are four combinations of observation and prediction results, as shown in Figure 2-2. True positive is when the observation is 1 and the model predicts it as 1; False positive is when the observation is 1 and the model predicts 0; False negative is when the observation is 0 and the model predicts it as 1; And false negative is when the observation is 0 and the model predicts it as 0.



Observed Predicted	True	False
True	True Positive	False Positive
False	False Negative	True Negative

Figure 2-2 Combination of observation and prediction.

Precision, recall, accuracy, and F1 score are mainly used as metrics for evaluating classification performance in various learning models. Precision is the percentage of actual true within the results that the model predicts as true. Recall is the percentage of the result that the model classifies as true within the observations is true. Precision and recall are calculated by Equation (2.3) and (2.4), where TP, FP, TN, FN expresses true positive, false positive, true negative, and false negative, respectively.

$$Precision = \frac{TP}{TP + FP} \tag{2.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.4}$$

Accuracy which is the most commonly used is the ratio of predicted cases correctly to all cases, as shown in equation (2.5). However, the accuracy is not appropriate to use when the data is imbalanced between classes. This is because, when there are majority classes that occupies most of the total data and minority classes with a relatively small proportion, the accuracy is calculated as a high value even if the minority class is not correctly predicted. Under the data imbalance state, the F1 score is a more appropriate metrics, since it considers performance of minor classes. The F1 score is defined as the harmonic average of precision and recall, as shown in equation (2.6). If either of the two values is low, the F1 score is calculated as low values. Therefore, both values must be appropriately high in order to obtain a high F1 score. In other words, it is a more proper indicator in data imbalance because the predicted result must be accurate in all classes to produce a high F1 score.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{2.5}$$

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2.6}$$

2.3 Bayesian Optimization

Bayesian optimization aims to find an optimal solution $x*$ that maximizes the unknown objective function f for the input x. Its main purpose is to quickly and effectively find the maximum value of f by sequentially examining the input value x with the minimum number of attempts.

2.3.1 Surrogate Model

Surrogate model refers to a model that makes a probabilistic estimation of an unknown objective function for investigated samples $\left(x_i,\ f(x_i)\right)(i = 1,2, \dots n)$. In general, the most commonly used probability model as a surrogate model is a *Gaussian process (GP)*. GP is a model to represent the probability distribution of the function to be estimated, and can be expressed through the mean function and the covariance function as shown in Equation (2.7) below. $\mu(x)$ and $k(x, x')$ express mean function and covariance function, respectively.

$$f(x) \sim GP\left(\mu(x),\ k(x,\ x')\right) \tag{2.7}$$

Figure 2-3 shows the process of Bayesian optimization through GP when the number of samples is 2, 3, and 4 in sequence. Dotted line represents the actual objective function, black solid line represents the average function of the irradiated points, and the purple region represents the standard deviation of the probability distribution of the objective function. As the investigated points are added, the purple region decreases because the probability estimation for the objective function becomes more accurate. The standard deviation is 0 at the point investigated, and as the distance from this point increases, the standard deviation increases, that is, the uncertainty increases.

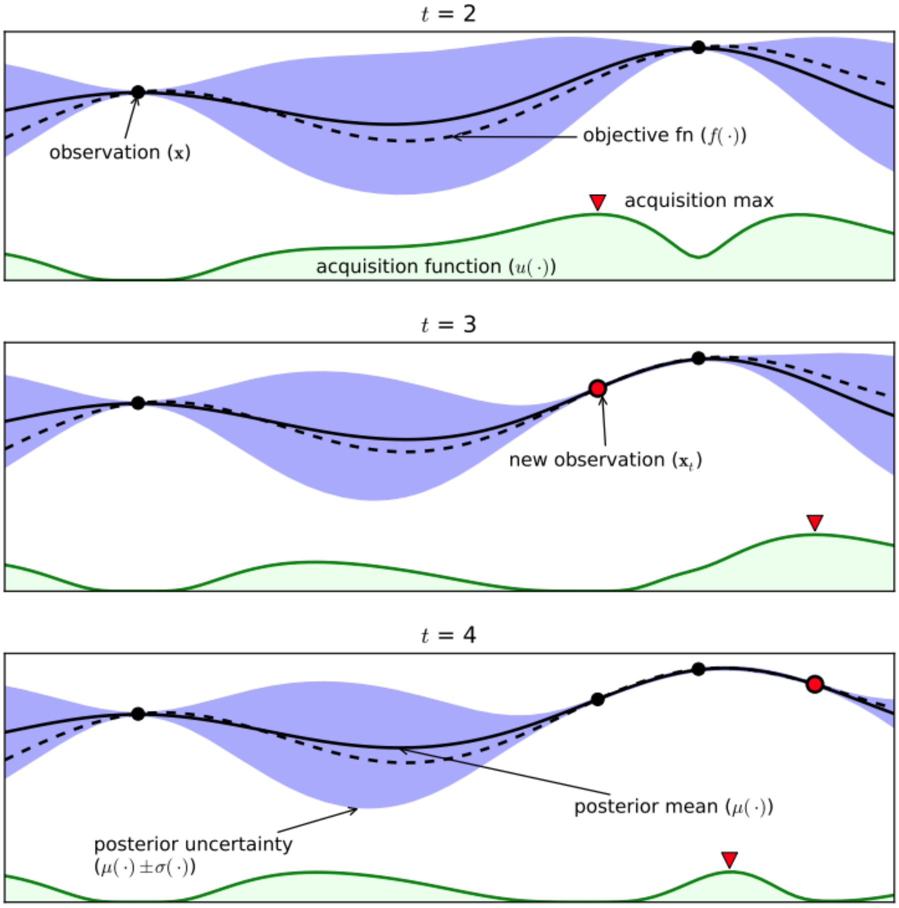


Figure 2-3 Bayesian optimization using GP [5].

2.3.2 Acquisition Function

Acquisition function refers to a function that recommends the next input candidate x_{n+1} based on $(x_1, f(x_1)), (x_2, f(x_2))\ldots$, and $(x_n, f(x_n))$ as a result of the investigation through the surrogate model so far. There are two strategies for recommending the most promising candidates for which the objective function has a maximum value. First, *Exploitation* is an estimation that the actual maximum value of the objective function exists around the maximum value in the sample candidates investigated so far. The second estimation strategy is *Exploration*. In the GP model, it was confirmed that as the distance from the investigated sample increases, the estimated standard deviation of the function increases. The larger the standard deviation is, the larger the range of variation of the objective function is, so there is a possibility that an actual maximum value may exist. In other words, the exploitation is to search for the area with the largest standard deviation in the estimated objective function. These two concepts are in a trade-off relationship with each other, and the next sample must be estimated by appropriately adjusting the relative strength of exploitation-exploration. The *Expected improvement (EI)* function as shown in Equation (2.8) is a function designed including the above two strategies, and is most often used as the acquisition function. $\mathit{\Phi}$ and ϕ represent the CDF and PDF of the standard normal distribution, respectively; ξ is a parameter that controls the relative intensity of exploitation and exploration.

$$EI(x) = E[\max(f(x) - f(x^{+}), 0]]$$

$$= \begin{cases} (\mu(x) - f(x^{+}) - \xi)\mathit{\Phi}(Z) + \sigma(x)\phi(Z) & if\ \sigma(x) > 0 \\ 0 & if \sigma(x) = 0 \end{cases} \tag{2.8}$$

$$Z = \begin{cases} \frac{(\mu(x) - f(x^{+}) - \xi)}{\sigma(x)} & if\ \sigma(x) > 0 \\ 0 & if\ \sigma(x) = 0 \end{cases} \tag{2.9}$$

2.4 STFT(Short-time Fourier transform)

Failure diagnosis is often difficult with simple time series characteristic factors such as mean, standard deviation, and kurtosis. Since the rotating body has a natural rotational frequency, changes due to failure often appear in the frequency domain. One of the most widely used techniques for converting time series data into the frequency domain is FFT (Fast Fourier Transform). However, although the FFT can inform the information in the frequency domain, it has a disadvantage of losing temporal information, so it is not suitable when data has a time varying property.

In order to compensate for the shortcomings of the FFT, short-time Fourier transform (STFT) is a technique that expresses both time domain information and frequency domain information. In order to obtain the STFT, DFT (Discrete Fourier Transform) is performed on each window while moving from the time series data to the desired window size at appropriate time intervals. DFT can be calculated by Equation (2.10), where x, X are time domain signal and frequency signal, respectively. In this way, the DFT values generated in each window can be made two-dimensional as a time axis. That is, the horizontal axis of time and the vertical axis of frequency generate 2D image data. STFT is a preprocessing technique that is frequently used for fault diagnosis because it includes both time domain and frequency domain information when the characteristics of data change over time.

$$\mathrm{X}\left(e^{jw}\right) = \sum_{n=-\infty}^{n=\infty} x[n]e^{-jwn} \tag{2.10}$$

$$\mathrm{x[n]} = \frac{1}{2\pi}\int_{2\pi} X(e^{jw})e^{jwn}dw \tag{2.11}$$

Chapter 3

Proposed Idea

3.1 Weight Balancing Parameters

As shown in Equation (2.1), the all data samples generally have equal weight, and the total loss function is calculated as the average of the loss values for each data sample. However, due to data imbalance, the learning model often does not properly train minor classes, and only learns major classes. In addition, while the model can easily classify data with clear differences between classes, it hardly classify some classes with similar physical characteristics. To solve this problem, it is necessary to learn more intensively the classes that the model does not classify well. By defining a new loss function by varying the weight for each data class, the degree of concentration of model training for each class can be adjusted. The transformed loss function is expressed as Equation (3.1), where w_i, m and n_i represent weight balancing parameter, the number classes, and the number of samples un each class, respectively.

$$L(\hat{Y}) = \sum_{i=1}^{m} [w_i \cdot \sum_{j=1}^{n_i} L_i \left(y_j, \hat{y}_j\right)] \qquad (3.1)$$

For the new loss function, the gradient descent of the neural network is expressed as equation (3.2). Since the model parameter θ is fed back by the weight balancing parameters and the learning rate, w_i and α become the hyper-parameters of the model. The learning rate α adjusts the overall scale of the gradient decent, and the weight parameter w_i adjusts the relative contribution of each class to the gradient decent.

In general, the value of the weight parameter for correcting the data imbalance is determined in inverse proportion to the number of samples for each data class for convenience. That is, small weight parameter values are assigned to a major classes with a large number of samples, and large weight parameter values are assigned to minor classes with a small number of samples. The equation for the conventional parameters are as in (3.3).

$$\begin{aligned} \theta &= \theta - \alpha \frac{\delta L}{\delta \theta} \\ &= \theta - \alpha \sum_{i=1}^{m} [w_i \cdot \frac{1}{\delta \theta} \sum_{j=1}^{n_i} L_i \left(y_j, \hat{y}_j \right)] \end{aligned} \tag{3.2}$$

$$w_i = \frac{1}{n_i} / \sum_{k=1}^{m} \frac{1}{n_k} (Conventional\ values) \tag{3.3}$$

3.2 Hyper-parameters Optimization using Bayesian Method

In order to classify motor defects, we can create the desired neural network structure, for given the data. When the previously defined weight balancing parameters and learning rate are set, the model trains the data by updating the model parameters using the gradient descent method in Equation (3.2). After training is completed, the model classification performance can be checked with the F1 score calculated as verification data. In other words, w_i and α become the input hyper-parameters, and the F1 score of the verification data becomes the objective function. The optimal values of hyper-parameters w_i and α that maximize the classification performance F1 score of a given model are found through Bayesian optimization.

Figure 3-1 shows the process of optimizing hyper-parameters through the Bayesian method. The given neural network trains the input data with initial hyper-parameters, and the trained model computes the F1 score with the verification data. Then, the F1 score according to the hyper-parameter becomes the initial point of the GP model. Next hyper-parameters to be investigated are recommended through the Bayesian method. Then, a new model is generated and training is conducted. This process is repeated, and Bayesian optimization is performed by maximizing the F1 score.

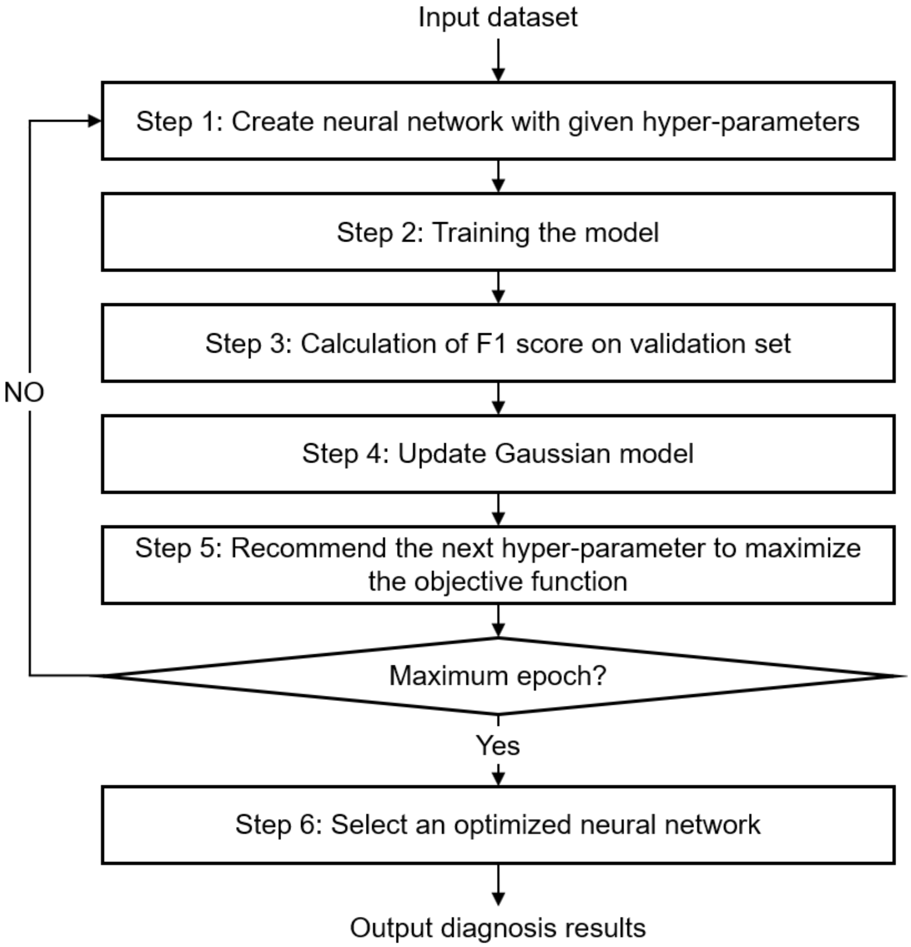

Input dataset
Step 1: Create neural network with given hyper-parameters
Step 2: Training the model
Step 3: Calculation of F1 score on validation set
Step 4: Update Gaussian model
Step 5: Recommend the next hyper-parameter to maximize the objective function
Maximum epoch?
NO
Yes
Step 6: Select an optimized neural network
Output diagnosis results

Figure 3-1 Flow chart of learning model update using Bayesian optimization.

Chapter 4

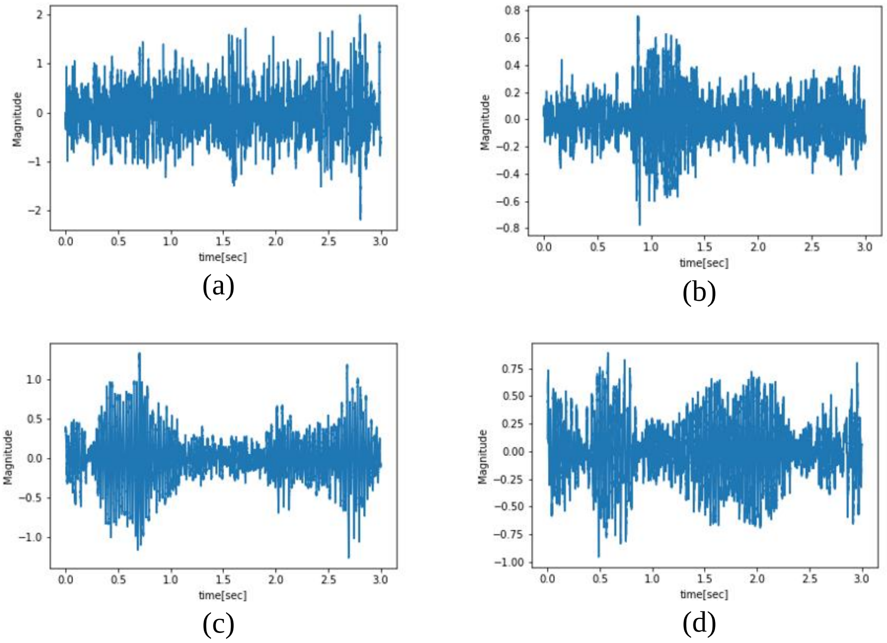
Experimental Process

4.1 Data Summary

The data is a single channel acoustic signal from a home washing machine motor. The data consists of a total of 257 sets, consisting of four types: normal data and three types of different defect data. There are 200 normal sets, 21 defect 1 sets, 30 defect 2 sets, and 6 defect 3 sets. One data set was acquired for 3 seconds with an acquisition frequency of 25600 [Hz] in a constant velocity section of 3000 to 3600 [RPM]. Information on the types of defects is given in Table 4-1.

Table 4-1 Defect types of motor.

	# of sets	**Description**
Normal	200	
Defect 1	21	Misalignment of motor axis
Defect 2	30	Foreign substance into motor bearing
Defect 3	6	Extraneous noise



(a)

(b)

(c)

(d)

Figure 4-1 Raw data examples: (a) normal, (b) defect 1, (c) defect 2, (d) defect 3.

4.2 Preprocessing

4.2.1 Sliding window filter

As mentioned earlier, the data in this study consists of a total of 257 sets, and each data set has a length of 76800 with an acquisition frequency of 25600 [Hz] for 3 seconds. Sliding window filter is a technique that extracts data of a desired length from long time series data by moving it at appropriate intervals, as shown in Figure 4-2. The sliding window filter was applied to increase the number of input data and reduce the shape of one input data. Data was extracted by moving as much as 2500 lengths with a window size of 5000 length per data set. 28 windows were created for each data set, resulting in a total of 7196(28*257) windows.

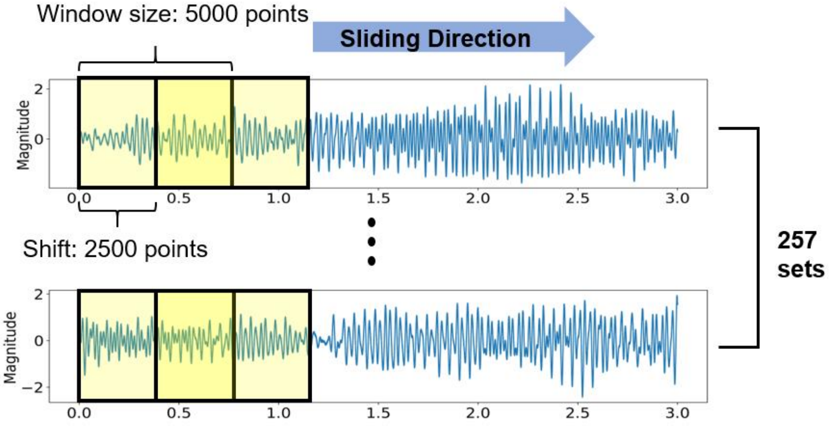
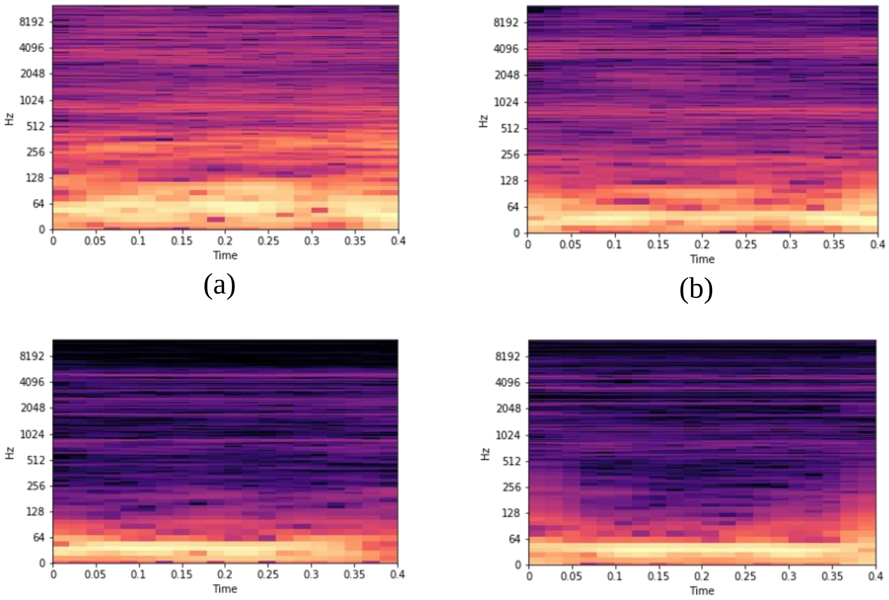


Figure 4-2 Sliding window filter.

4.2.2 STFT 2D Image

Since acoustic signal mainly has a characteristic that changes with time, STFT is often used as a preprocessing task [6]. As shown in Figure 4-1, the acoustic signal of this research is not constant over time and the variability is large. Therefore, although the input data is a signal acquired from a motor rotating at a constant speed, it is appropriate to apply STFT since it has a time-varying property. 7196 images are generated by applying STFT to each window obtained through the slicing window filter; The window size and hop size of the STFT are 2048 and 256, respectively. Figure 4-3 below shows an example of a 2D image applying STFT to the window extracted from each class.



(a) (b)

(c) (d)

Figure 4-3 STFT 2D image examples: (a) normal, (b) defect 1, (c) defect 2, (d) defect 3.

4.3 Neural Network

7196 windows were created through the sliding window filter, and 2D images were created by STFT of each window. As shown in Table 4-2, these 7196 images were divided into training, validation, and test data. The model was trained with training and validation data, and the model performance was confirmed with the test data.

Table 4-2 Composition of training, validation, and test data.

	Train	**Val**	**Test**	**Total**
Normal	3734	933	933	5600
Defect 1	392	98	98	588
Defect 2	560	140	140	840
Defect 3	112	28	28	168
Total	4798	1199	1199	7196

Since the input is a 2D STFT image, we choose 2D-CNN model as a neural architecture. As shown in Figure 4-4, the model was constructed using 3 convolution layers with a kernel size of 3*3, a stride size of 1*1, and zero-padding. The number of parameters of this configured model is 23796. The optimizer is Adam, and the loss function is categorical cross-entropy. The activation function of the last dense layer is SoftMax function. The batch size and learning rate were basically 16 and 0.0005, respectively.

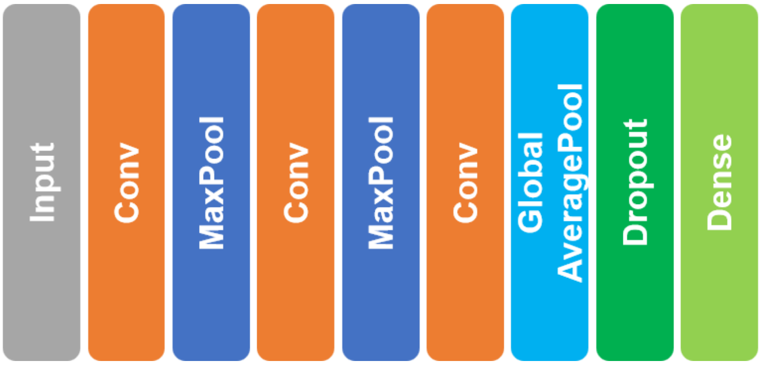


Figure 4-4 Neural architecture of 2D-CNN.

4.4 Optimization Setting

As explained in Part 3.2, Bayesian optimization retrieves the maximum value of the F1 score by sampling the next hyper-parameter candidates through GP after receiving the initial hyper-parameter. The hyper-parameters consist of a learning rate α and weight balancing parameters w_1, w_2, w_3, w_4 for four data classes. At this time, since the weight balancing parameters are relative ratios to each other, the parameter w_1 of the normal class is set to 1 and optimization is performed for α, w_2, w_3, and w_4. The weight parameters of the minor classes are optimized in log-scale in the range 1 to 1000, and the learning rate α is optimized in log-scale in the range of 0.0001 to 0.1.

For the initial parameters to be input to Bayesian optimization, w_1, w_2, w_3, w_4 are proportional to the reciprocal of the number of samples of each class as shown in equation (3.3) and w_2, w_3, w_4 were calculated by setting the criterion w_1 to 1. The learning rate a is set to 0.0005. That is, since the number of samples for each class are respectively 200, 21, 30, and 6, the calculated initial parameter (α,

w_1, w_2, w_3, w_4) is (0.0005, 1, 9.524, 6.667, 33.333). Bayesian optimization was set to 60 times under several trials.

4.5 Comparison Methods

This research proposes a method to increase the model's learning performance when there is data imbalance between classes. The preceding methods of solving data imbalance are largely divided into the data point of view and the algorithm point of view. In the data perspective, the data imbalance problem is alleviated by similarly controlling the number of input data for each class. As a comparison method for the proposed method in this study, the most commonly used under sampling and oversampling techniques were adopted. In both methods, the number of samples for each class is adjusted through preprocessing before data is input to the model. In the algorithm perspective, there are methods to solve data imbalance by modifying the model itself or changing the loss function or metric. In this paper, the technique using focal loss, which is frequently used recently, is adopted as a comparative control.

In this chapter, we introduce under sampling, over sampling and focal loss as comparison methods of the proposition, and explain how each method is applied to solve data imbalance.

4.5.1 Under Sampling

Under sampling is sampling by reducing the number of data in a major class. In order to avoid bias in data, in general, random sampling is performed in the major class similar to the level of the number of samples in the minor class. When performing random sampling in the normal class, if the number of samples is too small, the model cannot sufficiently train due to data loss. Therefore, 50 sets were randomly sampled from multiple classes, similar to the sum of the number of data sets of the other classes.

Table 4-3 # of data sets for each classes using under sampling

	Normal	**Defect 1**	**Defect 2**	**Defect 3**
# of data sets	50	21	30	6
# of 2D images	1400	588	840	168

4.5.2 Over Sampling

Oversampling is a technique of augmenting and sampling data in the minor class to have a similar number of samples to the major class. There are various methods to augment data. Random sampling may be performed by simply repeating the data, or data having a distribution similar to that of existing data can be created and sampled. In this study, the shift size of the sliding window filter is adjusted to increase the data differently for each class. If the shift size is reduced, a larger number of windows can be extracted from one data set, and thus data can be augmented and sampled more in the minor class. Table 4-4 shows the shift size for each class and the resulting window.

Table 4-4 Shift size and # of windows for each classes using oversampling

	Normal	**Defect 1**	**Defect 2**	**Defect 3**
# of data sets	200	21	30	6
Shift size	2500	1000	1000	200
# of windows	5600	1491	2130	2154

4.5.3 Focal Loss

Focal loss is a loss function created to mitigate data imbalance between classes. As shown in Equation (4.1), it is defined by multiplying the cross entropy function used as the loss function in the multiple classification problem by the term $(1-p_t)^\gamma$. Here, p, y, and γ means the prediction probability of the model, ground truth-label, and a focusing parameter that reshapes the loss function, respectively. Because of the multiplied term, training is performed by giving down-weight to classes that have already been classified well during back propagation. That is, the new loss function solve the data imbalance by emphasizing the training of difficult to classify data [9]. In this paper, we compare the classification results using the focal loss and the classification results using the proposed method from the viewpoint of an algorithmic approach.

$$
\begin{gathered}
FL(p,\, y) = -(1-p_t)^\gamma \log(p_t) \\
where\; p_t = \begin{cases} p & if\; y = 1 \\ 1-p & otherwise \end{cases}
\end{gathered}
\tag{4.1}
$$

Chapter 5

Result & Analysis

5.1 Classification Performance

After setting the weight balancing parameter and the learning rate as hyper parameters, Bayesian optimization is performed to find the optimal model that maximizes the F1 score of the verification data. This optimized model can check the classification performance using test data. The classification metric was confirmed by the F1 score and the loss function value for the test data.

The results were compared for the six cases as follows; ① When all classes are trained with the same weight, ② When class weights are adjusted by conventional parameters, ③ When hyper-parameters are optimized through Bayesian method. ④ Under sampling, ⑤ Over sampling, ⑥ Focal loss. For each case, the classification performance was calculated using the average value of five experiments.

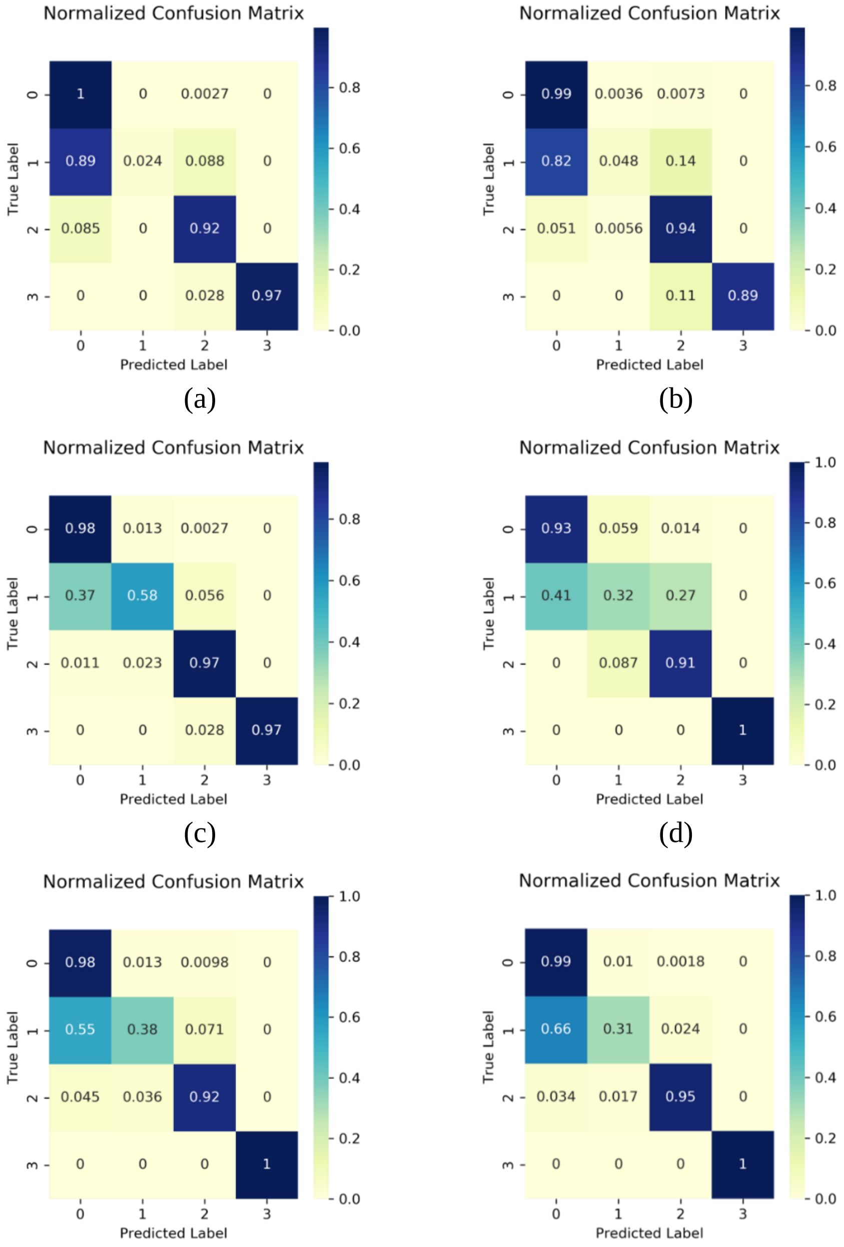
As shown in Table 5-1, it can be confirmed that there is no significant difference in classification performance in case ① and case ②. That is, the imbalance problem is not solved with the conventional parameters. However, in the case of optimization ③, the F1 score increased by about 0.05 and the loss function decreased near 0.1. The confusion matrix for each case is shown in Figure 5-1. Looking at cases ① and

②, it can be seen that defect 1 is hardly distinguished from normal. However, in case ③, it can be seen that about 60% of defect1 is distinguished from normal.

In both sampling methods(case ④, ⑤), it can be seen that the classification performance of defect 1 is slightly increased, but the overall performance is decreased. This is because although the sampling was adjusted for each class to reduce the imbalance rate, the classification performance in minor classes did not improve much. However, when focal loss was used (⑥), both the classification performance of the minor class and the overall F1 score increased. In particular, it was confirmed that the classification performance of defect 1 increased by about 30%. In other words, focal loss can effectively solve data imbalance than the existing loss function. In conclusion, the proposed method is superior to the existing methods in terms of classification performance of minority classes and overall classification performance.

Table 5-1 Classification metrics for each case.

	①	②	③	④	⑤	⑥
Mean of F1 score	0.901	0.904	0.946	0.827	0.879	0.917
Mean of loss	0.287	0.270	0.161	0.410	0.280	1.215



(a) (b) (c) (d) (e) (f)

Figure 5-1 Normalized confusion matrix: (a) case ①, (b) case ②, (c) case ③, (d) case ④, (e) case ⑤, (f) case ⑥.

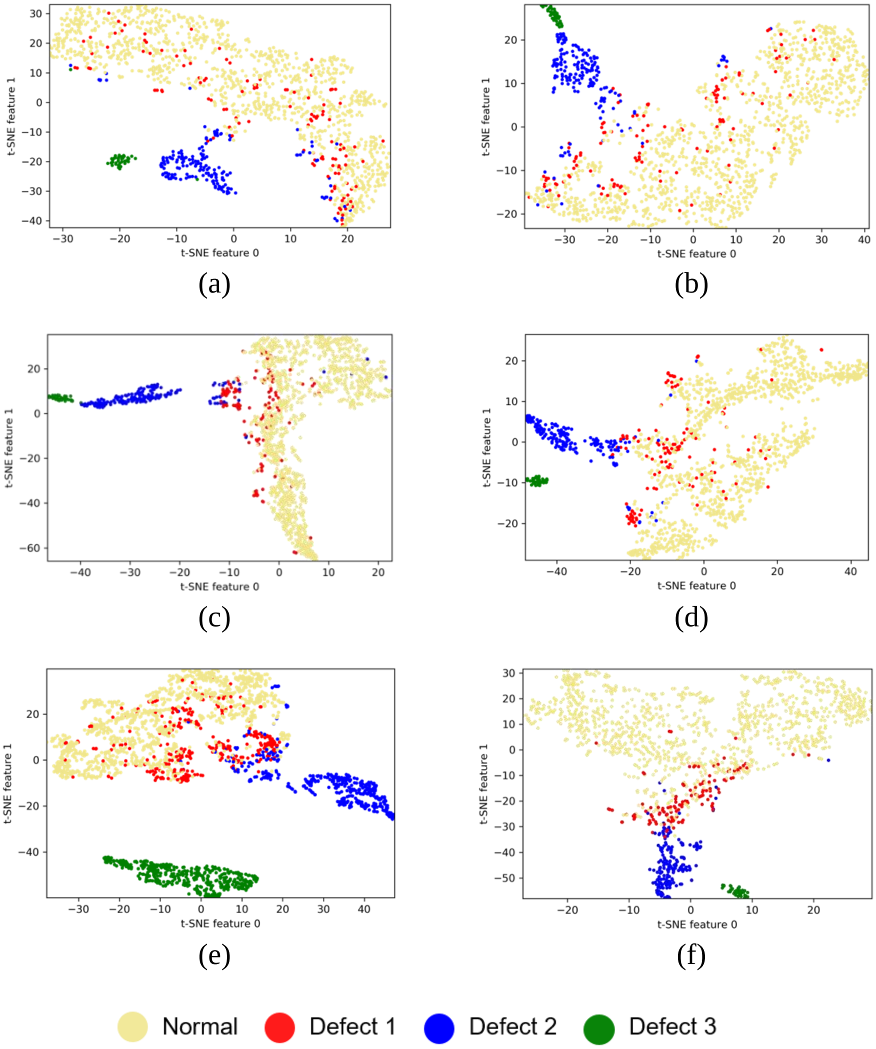
5.2 Feature Visualization

Figure 5-2 shows the results of feature visualization through t-SNE for each case of the previously classified results. This is the result by reducing the features that appeared before the last dense layer of the 2D-CNN model to 2D dimension.

Looking at cases ① and ②, it can be seen that defect 2 and defect 3 are well separated from other classes, but defect 1 is hardly classified with normal. However, in case ③ of the proposed method, it can be seen that defect 1 can be distinguished from normal, and other types of defects are clearly separated.

In cases ④ and ⑤, it can be seen that the scale difference is reduced through the sampling method by looking at the number of samples for each class. As discussed earlier, it can be seen that the points of defect 1 are more densely gathered and distinguished from normal to some extent. However, it still seems that the classification performance is insufficient for fault diagnosis.

Finally, looking at case ⑥, similar to case ③, defect 1 can be distinguished from normal. Although the F1 score is lower than the proposed method, it performs much better than other data imbalance solutions.



(a) (b)

(c) (d)

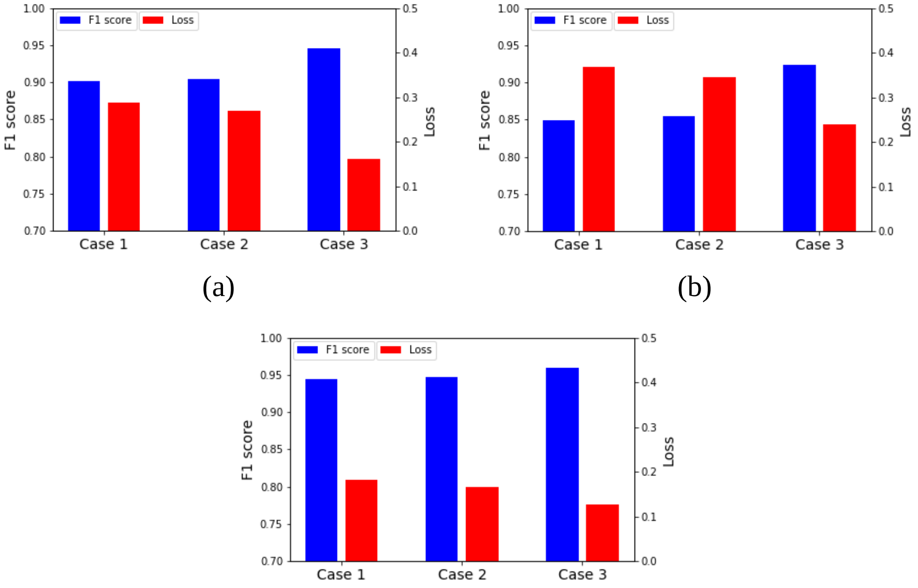
(e) (f)

Figure 5-2 Feature visualization through t-SNE: (a) case ①, (b) case ②, (c) case ③, (d) case ④, (e) case ⑤, (f) case ⑥.

5.3 Application for Different Imbalance Level

This study improves classification performance when there is data imbalance between classes. The experiment was conducted when normal is 200, defect 1 is 21, defect 2 is 30, and defect 3 is 6, and it can be seen that there is a data imbalance in the normal is significantly more than the defect. Following the previous experiment, in order to verify the performance of the proposed method at different imbalance levels, we applied this proposed method when the number of normal data is 100 or 400. The case of 100 was named low imbalance level, and the case of 400 was named high imbalance level, and the case of 200 was named intermediate imbalance level. The cases ①, ② and ③ were compared, as shown in the figure 5-3.

In all three imbalance levels, it was confirmed that in case ③, the F1 score was the highest and the loss was the lowest. That is, it can be said that the proposed method is applicable to various levels of imbalance dataset.



(a)

(b)

(c)

Figure 5-3 F1 score and loss for each cases: (a) Intermediate imbalance level, (b) low imbalance level, (c) high imbalance level.

5.4 Comparison with Other Hyper-parameter Optimization Methods

There are many ways to optimize hyper-parameters in deep learning, but there are four commonly used methods; Manual search, grid search, random search, and Bayesian optimization. Manual search is Method of relying on intuition to directly search for the optimal hyper-parameters value. For grid search, candidate of hyper-parameters are selected at regular intervals within the search region. In random search, hyper-parameters within the search section are selected through random sampling. Finally, Bayesian optimization is a methodology for selecting the next hyper-parameter candidate by reflecting the previously searched knowledge.

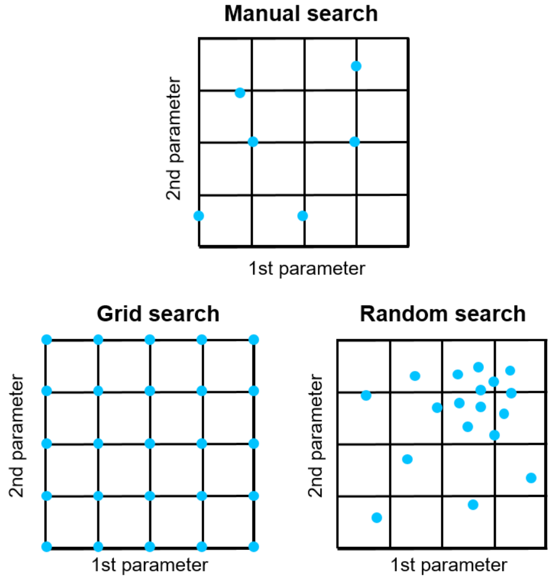
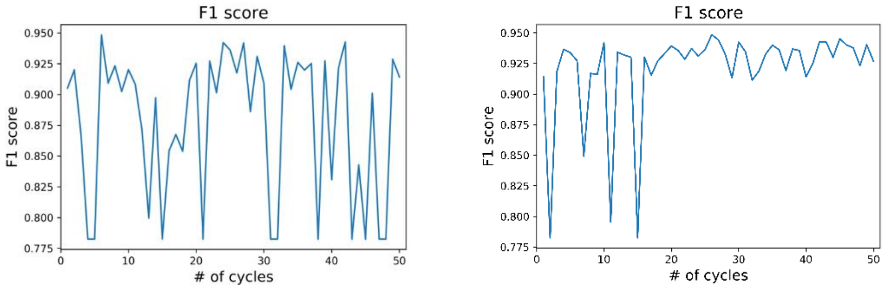


Figure 5-4 Types of hyper-parameter optimization methods.

In this section, we compare random search and Bayesian optimization, applying in the experiment. In both methods, if the number of searches is large enough, the classification performance of the model is similarly improved. Since the neural network is fixed, it can be analyzed that the classification performance of the model converges to a certain threshold even when the optimal value of the weight balancing parameters is found through the search. In Figure 5-5, it can be seen that through random search, the F1 score continuously vibrates even when the number of searches increases. In other words, it is impossible to find a meaningful search section with

50 times. On the other hand, through Bayesian optimization, the F1 score tends to converge after 20 times. That is, since sections that do not need to be searched are removed and the next sections are searched, the optimum value can be more efficiently found.



(a) (b)

Figure 5-5 F1 score vs epochs for each optimization method: (a) Random search (b) Bayesian optimization.

Chapter 6

Conclusion

This study proposes a method to solve the data imbalance problem by defining the weight parameter in the loss function for each class and optimizing the hyper-parameter through the Bayesian method in deep learning based motor fault diagnosis. Through the case study, it was confirmed that the proposed method improves the classification performance of minor classes in the given model and data. In addition, it was confirmed that it shows better performance than other existing methods of solving data imbalance.

The contribution of this study can be summarized in three ways. First, the paper proposes framework of solving data imbalance using Bayesian optimization of weight balancing parameters. Secondly, this method is applicable regardless of data type, model type, and preprocessing. Finally, optimized model is found for given data & model shape, improving classification performance of minor classes.

In future work, this study will be explored for other datasets and it will prove it can be applicable for various fields. Furthermore, since the classification performance of minority classes has not yet been completely improved, other attempts are required to compensate for this.

References

[1] Shao, Si-Yu, et al. "A deep learning approach for fault diagnosis of induction motors in manufacturing." Chinese Journal of Mechanical Engineering 30.6 (2017): 1347-1356.

[2] Wang, Shoujin, et al. "Training deep neural networks on imbalanced data sets." 2016 international joint conference on neural networks (IJCNN). IEEE, 2016.

[3] Sudre, Carole H., et al. "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations." Deep learning in medical image analysis and multimodal learning for clinical decision support. Springer, Cham, 2017. 240-248.

[4] Snoek, Jasper, Hugo Larochelle, and Ryan P. Adams. "Practical bayesian optimization of machine learning algorithms." Advances in neural information processing systems 25 (2012): 2951-2959.

[5] Brochu, Eric, Vlad M. Cora, and Nando De Freitas. "A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning." arXiv preprint arXiv:1012.2599 (2010).

[6] Liu, Hongmei, Lianfeng Li, and Jian Ma. "Rolling bearing fault diagnosis based on STFT-deep learning and sound signals." Shock and Vibration 2016 (2016).

[7] Adam Glowacz, Zygfryd Glowacz. “Diagnosis of stator faults of the single-phase induction motor using acoustic signals,” Applied Acoustics, 2017

[9] Zhiqiang Chen, et al., “Deep neural networks-based rolling bearing fault diagnosis,” Microelectronics Reliability, 2017

[9] Pasupa, Kitsuchart, Supawit Vatathanavaro, and Suchat Tungjitnob. "Convolutional neural networks based focal loss for class imbalance problem: A case study of canine red blood cells morphology classification." Journal of Ambient Intelligence and Humanized Computing (2020): 1-17.

[10] Tran, Giang Son, et al. "Improving accuracy of lung nodule classification using deep learning with focal loss." Journal of Healthcare Engineering 2019 (2019).

[11] Liang, Xiao. "Image-based post-disaster inspection of reinforced concrete bridge systems using deep learning with Bayesian optimization." Computer-Aided Civil and Infrastructure Engineering 34.5 (2019): 415-430.

[12] Douzas, Georgios, and Fernando Bacao. "Effective data generation for imbalanced learning using conditional generative adversarial networks." Expert Systems with applications 91 (2018): 464-471.

국문 초록

베이지안 기반 데이터 클래스 가중치 최적화를 통한 딥러닝 모터 결함 진단

모터는 산업용 로봇, 가정용 기기, 교통 수단 등 다양한 분야에 사용되어 지고 있고 모터 부품 하나의 결함이 전체 기계 시스템의 고장까지 야기할 수 있기 때문에, 모터 결함(Motor defect) 진단은 필수적이다. 이에, 최근 딥러닝을 사용한 데이터 기반 접근 방법이 고장 진단 연구에 많이 적용되어지고 있다. 하지만 실제 산업 환경에서는 고장 사례가 많이 나타나지 않기 때문에 고장 데이터들이 정상 데이터에 비해 많이 부족하다. 이러한 데이터 불균형(Data imbalance)은 뉴럴 네트워크 모델이 고장에 관한 정보를 충분히 학습을 불가능하게 하여, 딥러닝 기반 알고리즘의 고장 진단 성능을 현저히 떨어뜨린다. 본 연구는 딥러닝 모델이 데이터를 학습할 시, 손실 함수(Loss function)에서 데이터 클래스별 가중치 균형 파라미터(Weight balancing parameter)를 정의하여 데이터 불균형 문제를 해결하고자 한다. 클래스별 가중치 파라미터를 조절함으로써, 소수 클래스의 분류 성능을 향상시키는데 기여하고자 한다. 또한 베이지안 최적화(Bayesian optimization) 방법을 통해, 최적화된 가중치 파라미터를 가지는 학습 모델을 찾을 수 있다. 본 연구의 실험 결과를 통해, 최적화된 학습 모델이 기존 모델에 비해 다수 및 소수 클래스에서 모두 향상된 분류 성능을 보임을 확인할 수 있다. 즉, 클래스별 불균형 조절을 위한 입력 데이터의 편집 없이 모델을 학습시킬 수 있다

주요어: 모터 결함
데이터 불균형
가중치 균형 파라미터
손실 함수
베이지안 최적화

학 번: 2019-22241