ISSN:1738-6764, eISSN:2093-7504

Controlling Neural Network Training via Modification of Cross-Entropy Loss Function

Sang-Hoon Oh [1,*]

[1] Mokwon University; Professor; ohsanghoon16@gmail.com
* Correspondence

https://doi.org/10.5392/IJoC.2025.21.4.133

Manuscript Received 25 July 2025; Received 6 November 2025; Accepted 11 November 2025

Abstract: *There have been various loss functions proposed to improve the training of neural networks with sigmoid activation output nodes. For neural networks with softmax activation output nodes, the cross-entropy loss function is commonly used for training and several attempts have been made to improve the performance of such networks by modifying the standard cross-entropy loss. However, rather than simply aiming to improve overall classification accuracy, it is often necessary to address misclassification costs differently depending on their real-world importance. In practice, the cost of errors can vary greatly across domains such as finance, security, insurance, and healthcare. From this perspective, this paper proposes a modified cross-entropy loss function designed to control the training of neural networks with softmax output nodes. The effectiveness of the proposed loss function is demonstrated through simulations on the CEDAR handwritten digit recognition task, showing that classification performance can be adjusted according to the order of the modified loss function. This approach will serve as the basis for designing a novel cost-sensitive learning method tailored to neural networks with softmax outputs.*

Keywords: Error Back-propagation; Cross-entropy Loss Function; Cost-sensitive Learning; Neural Networks; Softmax Activation

1. Introduction

It is well known that feed-forward neural networks with multiple hidden layers are fundamental models of machine learning, and they are typically trained by the EBP (Error Back-Propagation) algorithm [1, 2]. Moreover, based on the theoretical proof that feed-forward neural networks with sufficient hidden nodes are universal approximators of arbitrary functions [3], numerous efforts have been made to achieve satisfactory performance on real-world tasks. To train neural networks with sigmoidal activation outputs, the EBP algorithm minimizes the MSE (mean-squared error) function between the actual and desired outputs [1]. However, there were many reports that the EBP algorithm often exhibits slow convergence and poor generalization to test samples [4-7].

To address these issues, various approaches have been proposed to modify or replace the objective function, aiming to improve the performance of the EBP algorithm by employing alternative loss functions beyond MSE [8-15]. For example, additive noise was introduced into the target signals to enhance MSE-based learning [11]. The MLS (mean-log square) loss function was proposed to suppress the excessive weight updates caused by outliers of training data [12]. Ooyen and Nienhuis introduced the binary cross-entropy loss function to accelerate the learning convergence by alleviating incorrect saturation of output nodes [13]. However, the binary cross-entropy loss function method is prone to overspecialization to training data [14]. To address this limitation, the *n*-th order extension of the binary cross-entropy error has been shown to yield improved performance by mitigating overspecialization [14]. Another notable approach is the classification figure of merit (CFM) objective function, proposed by Hampshire and Waibel, which also aims to reduce overspecialization to training data [15].

In contrast, for multi-class classification problems-where only one class is correct-the softmax function is commonly used as the output activation in neural networks, as it produces a well-defined probability distribution

over all classes and facilitates direct comparison of class probabilities [2], [16]. Neural networks with softmax-activated output nodes are typically trained by minimizing the cross-entropy loss function [16-18]. To further improve classification performance, several approaches have been proposed that modify the standard cross-entropy loss function [19-22]. For example, to enhance classification tasks involving one-hot encoded class labels, Shim proposed incorporating an additional term that depends on the predicted probability of the ground-truth class [19]. The MPCE (Maximum Probability-based Cross Entropy) loss function was introduced to exploit the maximum predicted probability across all classes in the output distribution [20]. To mitigate the vanishing gradient issue that arises when the cross-entropy loss becomes nearly zero, Li et al. introduced a regularization term into the standard cross-entropy formulation. This term penalizes the probability assigned to incorrect classes relative to the ground-truth class [21]. Additionally, Ho and Wookey proposed the real-world-weight cross-entropy loss function in both binary and single-label multi-class classification variants [22].

However, in practice, misclassification can incur vastly different consequences depending on the domain, particularly in areas like finance, security, insurance, and healthcare. For example, in medical diagnosis, the consequence of misclassifying a brain tumor as a normal cell can be far more severe and potentially fatal than erroneously diagnosing a normal cell as a brain tumor. In the context of financial fraud detection, the cost of failing to identify a fraudulent transaction-by classifying it as legitimate-is significantly higher than the cost of mistakenly classifying a legitimate transaction as fraudulent. In the domain of security, misclassifying a cyberattack as benign traffic can lead to serious breaches, data loss, or system compromise, whereas a false alarm may only cause temporary inconvenience or additional verification effort. Therefore, it is often necessary to address misclassification costs differently depending on their real-world importance rather than simply aiming to improve overall classification accuracy. To achieve this goal, the learning performance of the neural network should be controlled through various means, such as modifying the loss function.

A modified version of the binary cross-entropy loss function has been proposed to regulate the learning behavior of neural networks with sigmoid outputs, based on the relative importance of the class to which the input data belongs [23-25]. In this approach, the order parameter of the loss function controls the error signal of the output nodes, following an *n*-th order relationship between the desired and actual output values [23-26]. Specifically, when data from a more important class is presented, the network undergoes more intensive training with larger weight updates; conversely, data from less important classes leads to more conservative training with smaller weight updates. These approaches represent a form of cost-sensitive learning designed to address class imbalance problems.

Building on this perspective, this paper proposes a modified cross-entropy loss function for neural networks with softmax output nodes, which incorporates an order parameter to adjust the magnitude of weight updates. Specifically, by tuning the order of the proposed loss function, the training dynamics of softmax-based neural networks can be effectively controlled. This approach offers a foundation for developing a new cost-sensitive learning algorithm for neural networks with softmax output nodes.

In section 2, we provide a brief overview of neural networks with softmax output nodes and the use of the cross-entropy loss function for training. We then introduce a modified version of the loss function designed to control the training process. In section 3, the effectiveness of the proposed method is demonstrated through simulations on the CEDAR handwritten digit recognition dataset. Finally, section 4 presents the conclusions of the paper.

2. Modified Cross-Entropy Loss Function

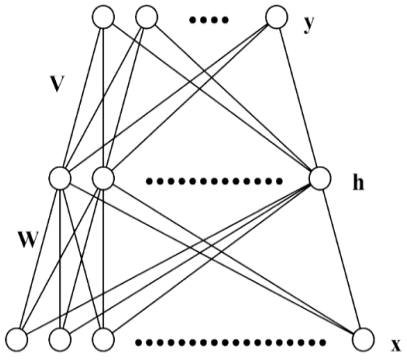


Figure 1. The architecture of a feed-forward neural network.

Let us consider a feed-forward neural network has N inputs, H hidden nodes, and M output nodes. Figure 1 illustrates the architecture of the network. When an input data $\boldsymbol{x} = [x_1, x_2, \dots, x_N]^T$ is fed into the neural network, the activation of the *j*-th hidden node is given by

$$h_j = f(\hat{h}_j) = f\left(w_{j0} + \sum_{i=1}^{N} w_{ji} x_i\right),\ j = 1,2,\dots,H, \tag{1}$$

where $f(\cdot)$ denotes the activation function of the hidden nodes, w_{ji} is the weight connecting input x_i to hidden node h_j, w_{j0} is the bias term of h_j, and $\hat{h}_j$ is the net input or the weighted sum to the hidden node. The activation of the *k*-th output node is computed using the softmax function:

$$y_k = \frac{e^{\hat{y}_k}}{\sum_{j=1}^{M} e^{\hat{y}_j}}\ \ (k = 1,2,\dots,M), \tag{2}$$

where the net input $\hat{y}_k$ to the output node is defined as

$$\hat{y}_k = v_{k0} + \sum_{j=1}^{H} v_{kj} h_j. \tag{3}$$

Here, v_{k0} is the bias of output node y_k, and v_{kj} denotes the weight connecting hidden node h_j to output node y_k.

The desired output vector $\boldsymbol{t}$ corresponding to a training sample $\boldsymbol{x}$ is one-hot encoded as follows:

$$t_k = \begin{cases} 1, & \text{if } \boldsymbol{x} \text{ originates from class } k \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

As a distance measure between the actual and desired outputs, the cross-entropy loss function is typically used [16]:

$$E_{CE} = -\sum_{k=1}^{M} t_k\ log\ y_k. \tag{5}$$

To minimize E_{CE} over the training dataset, the weights v_{kj} are iteratively updated using the gradient descent method:

$$\Delta v_{kj} = -\eta \frac{\partial E_{CE}}{\partial v_{kj}} = \eta \delta_k^{(output)} h_j, \tag{6}$$

where the error signal for the output node is

$$\delta_k^{(output)} = -\frac{\partial E_{CE}}{\partial \hat{y}_k} = t_k - y_k, \tag{7}$$

and η is the learning rate. Similarly, the weights w_{ji} are updated as follows:

$$\Delta w_{ji} = -\eta \frac{\partial E_{CE}}{\partial w_{ji}} = \eta \delta_j^{(hidden)} x_i = \eta x_i f'(\hat{h}_j) \sum_{k=1}^{M} v_{kj}\, \delta_k^{(output)}. \tag{8}$$

This weight-update procedure corresponds to the well-known error back-propagation (EBP) algorithm for minimizing the standard cross-entropy loss function [16-18]. As shown in Eqs. (6) and (7), the weight updates are proportional to the error signal of the output node, which is the difference between the desired and actual outputs.

To provide greater flexibility in controlling the learning process, we propose a modification of the cross-entropy loss function in which the error signal at the output node is formulated as an *n*-th order function of the difference between the desired and actual outputs. Under the assumption that the input sample belongs to class *k*, the proposed loss function is defined as:

$$E_{mod} = -[\sum_{j \neq k} t_j\ log\ y_j + \int \frac{(2t_k - 1)^{n+1} (t_k - y_k)^n}{y_k (1 - y_k)} dy_k], \tag{9}$$

where *n* is the order of the modification and *k* is the index of the true class label for the input data. For softmax output nodes,

$$\frac{\partial y_k}{\partial \hat{y}_j} = \begin{cases} y_k(1 - y_k), & \text{if } j = k \\ -y_k y_j, & \text{otherwise.} \end{cases} \tag{10}$$

Also,

$$\frac{\partial}{\partial \hat{y}_k} [\int \frac{(2t_k - 1)^{n+1} (t_k - y_k)^n}{y_k (1 - y_k)} dy_k] = \frac{(2t_k - 1)^{n+1} (t_k - y_k)^n}{y_k (1 - y_k)} \frac{\partial y_k}{\partial \hat{y}_k}, \tag{11}$$

Therefore, the resulting error signal for the target output node y_k is

$$\delta_k^{(output)} = -\frac{\partial E_{mod}}{\partial \hat{y}_k} = -y_k \sum_{j \neq k} t_j + (2t_k - 1)^{n+1}\left(t_k - y_k\right)^n. \quad (12)$$

Assuming the input sample belongs to class *k*, we have $t_k = 1$ and $t_j = 0$ $(j \neq k)$, which simplifies Eq. (12) to

$$\delta_k^{(output)} = \left(1 - y_k\right)^n. \quad (13)$$

For a non-target output node $y_i($ where $i \neq k)$, the error signal is:

$$\delta_i^{(output)} = -\frac{\partial E_{mod}}{\partial \hat{y}_i} == -y_i \sum_{j \neq k} t_j - y_i \frac{(2t_k - 1)^{n+1}\left(t_k - y_k\right)^n}{1 - y_k} = -y_i\left(1 - y_k\right)^{n-1}. \quad (14)$$

When the order *n* is equal to 1, Eqs. (13) and (14) reduce to Eq. (7), indicating that the proposed loss function is indeed a generalization of the standard cross-entropy loss.

We can further analyze the characteristics of the proposed loss function by visualizing the error signals using two-dimensional and three-dimensional plots. As shown in Figure 2, which depicts $\delta_k^{(output)}$ for various values of *n*, the update strength can be controlled through the order *n*, with weight updates remaining proportional to the error signal magnitude. Figure 3 presents a 3D plot of $\delta_i^{(output)}(i \neq k)$ when *n*-th 4. When $y_k = 0$, $\delta_i^{(output)} = -y_i$, which matches the behavior under the cross-entropy loss. And, when $y_k = 1$, $\delta_i^{(output)} = 0$. For $0 < y_k < 1$, the error signal behaves as $\delta_i^{(output)} = -y_i(1 - y_k)^{n-1}$, where the slope of the curve $\delta_i^{(output)}$ versus y_i is governed by the term $-(1 - y_k)^{n-1}$.

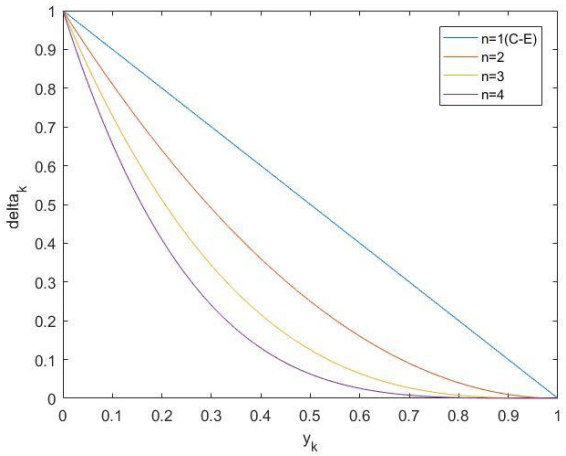


Figure 2. Plots of the error signal for the target node, $\delta_k^{(output)}$. Here, *n* is the order of the proposed loss function.

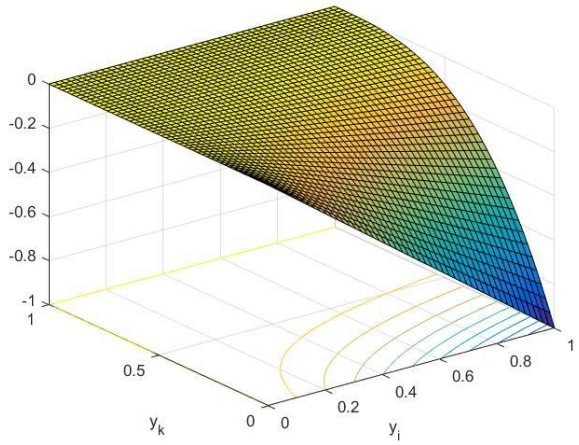
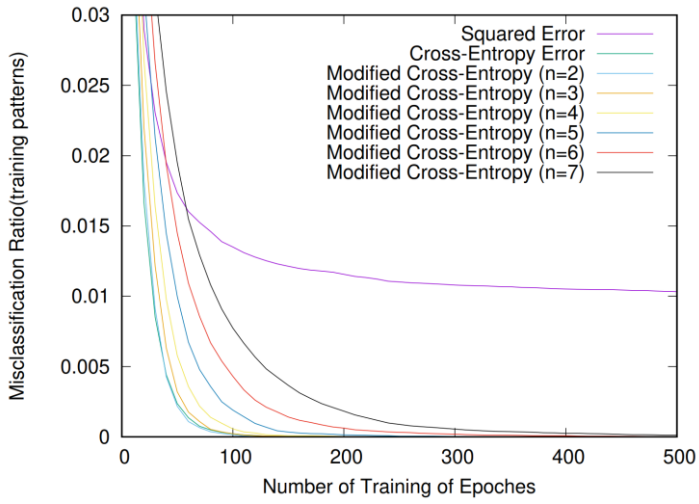


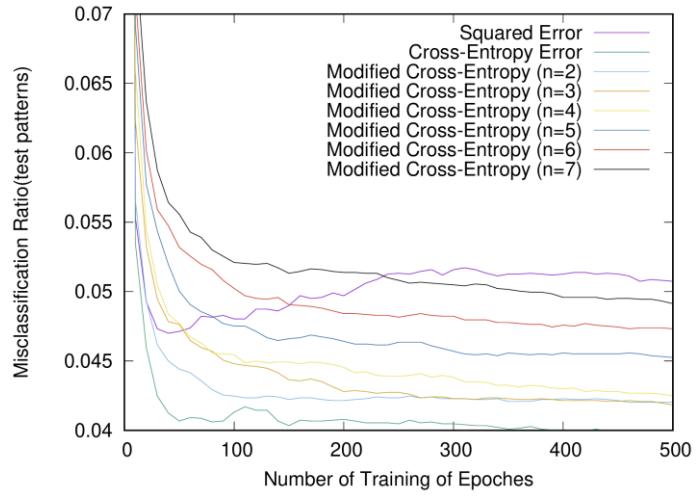
Figure 3. Plots of the error signal for non-target nodes, $\delta_i^{(output)}$ when *n*=4.

3. Simulations

To verify the effectiveness of the proposed loss function, we conduct simulations on the CEDAR handwritten digit recognition task [27]. Each digit image consists of 12×12 pixels, with each pixel taking integer values from 0 to 15. The feed-forward neural network comprises 144 input nodes, 60 hidden nodes with the sigmoid activation function, and 10 output nodes with the softmax activation function. A total of 18,468 handwritten digit images are used for training, and 2,213 images are used for testing. Since a fair comparison cannot be ensured when using identical learning rates, we instead derive the learning rates such that the expected value of the product of the target node error term and the learning rate, $E\left\{\eta\delta_k^{(output)}\right\}$, is the same across simulations, assuming that y_k follows a uniform distribution over [0,1] [14]. That is, we used $\eta_{CE} = 0.01$ for the cross-entropy loss and $\eta_{mod} = \frac{n+1}{2}\eta_{CE}$ for the modified loss.



(a)



(b)

Figure 4. Simulation results for the handwritten digit recognition task. "Squared Error" refers to EBP learning using the mean-squared error function, while "Modified Cross-Entropy" represents the proposed learning algorithm with the order parameter *n*. "Cross-Entropy" corresponds to the modified cross-entropy with $n = 1$. **(a)** Misclassification ratio for the training patterns; **(b)** Misclassification ratio for the test patterns.

We simulate EBP training to minimize the proposed loss function with the order parameter *n* ranging from 1 to 7. For each value of *n*, nine simulations are conducted using different initializations of the feed-forward neural networks, where the initial weights are randomly drawn from a uniform distribution over $[-1\times10^{-4}, 1\times10^{-4}]$. The results are averaged to produce the figures. Figure 4(a) presents the misclassification ratio for the training patterns. As the order parameter *n* increases, the magnitude of the output error term $\delta_k^{(output)}$ decreases, which in turn slows down the convergence of learning. Consequently, as shown

in Figure 4(b), the misclassification ratio for the test patterns increases. This demonstrates that the learning performance can be controlled by adjusting the order parameter *n*.

Additionally, we simulate a feed-forward neural network with sigmoidal output activation functions trained to minimize the MSE function. Here, we used $\eta_{MSE} = 0.06$. As expected, the performance of the MSE-based model is inferior to those trained using the cross-entropy and the proposed modified cross-entropy loss functions [17]. This is because neural networks with softmax output nodes are generally more suitable for classification tasks than those with sigmoid outputs, and the cross-entropy loss is more effective than MSE in such scenarios. For clarity, we summarize training parameters in Table 1.

Although the effectiveness of the proposed loss function has been verified through simulations on the CEDAR handwritten digit dataset, further experiments on additional datasets are necessary to provide stronger empirical support for the claim. Therefore, we plan to conduct experiments on healthcare datasets, such as thyroid and mammography disease classification tasks, to verify whether the proposed loss function can enhance classification performance in real-world disease prediction scenarios.

Table 1. Summary of simulation parameters.

Architecture	Weight Initialization	Learning Rate	Training Epoch
144 inputs, 60 sigmoidal hidden nodes, 10 softmax output nodes	Uniform distribution over $[-1 \times 10^{-4}, 1 \times 10^{-4}]$	MSE(sigmoid output): $\eta_{MSE} = 0.06$ CE: $\eta_{CE} = 0.01$ Modified Method: $\eta_{mod} = \frac{n+1}{2}\eta_{CE}$	500 epochs

4. Conclusions

Various loss or objective functions have been proposed to improve the performance of neural networks with sigmoid activation output nodes, beyond the commonly used MSE function. Additionally, numerous modifications of the cross-entropy loss function have been introduced to enhance the performance of neural networks with softmax output nodes. However, improving overall classification accuracy alone is often insufficient. Misclassification costs must be addressed differently depending on their importance in specific real-world applications. These costs can vary significantly across domains such as finance, security, insurance, and healthcare.

To this end, we proposed a modification of the standard cross-entropy loss function to control neural network learning behavior using an order parameter embedded in the loss function. Two-dimensional and three-dimensional plots of the error signal demonstrate that the proposed loss function generates an error signal of the target output node, which follows an *n*-th order function of the difference between the actual and desired outputs.

We validated the effectiveness of the proposed loss function through simulations on the CEDAR handwritten digit recognition task. As expected, the order parameter controlled the network's training dynamics—higher values of the order parameter led to slower convergence during learning and, consequently, an increase in the misclassification ratio for the test patterns. Based on this approach, a novel cost-sensitive learning algorithm can be devised for neural networks with softmax outputs.

Conflicts of Interest: The author declares no conflict of interest.

References

[1] D. E. Rumelhart and J. L. McClelland, Parallel Distributed Processing, MIT Press, Cambridge, MA, 1986, doi: https://doi.org/10.7551/mitpress/5236.001.0001.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998, doi: https://doi.org/10.1109/5.726791.

[3] K. Hornik, M. Stinchcombe, and H. White, "Multilayer Feedforward Networks Are Universal Approximators," Neural Networks, vol. 2, pp. 359-366, 1989, doi: https://doi.org/10.1016/0893-6080(89)90020-8.

[4] R. K. Cheung, I. Lustig, and A. L. Kornhauser, "Relative effectiveness of training set patterns for backpropagation," Proc. IJCNN, Washington, D.C., vol. I, pp. 673-678, Jan. 15-19, 1990.

[5] Y. Lee, S. H. Oh, and M. W. Kim, "An analysis of premature saturation in backpropagation learning," Neural Networks, vol. 6, pp. 719-728, 1993, doi: https://doi.org/10.1016/S0893-6080(05)80116-9.

[6] J. R. Chen and P. Mars, "Stepsize variation methods for accelerating the backpropagation algorithm," Proc. IJCNN, Washington, D.C., vol. I, pp. 601-604, Jan. 15-19, 1990.

[7] A. Rezgui and N. Tepedelenlioglu, "The effect of the slope of the activation function on backpropagation algorithm," Proc. IJCNN, Washington, D.C., vol. I, pp. 707-710, Jan. 15-19, 1990.
[8] S.-H. Oh, "Statistical Analyses of Various Error Functions for Pattern Classifiers," Proc. Convergence on Hybrid Information Technology, Daejon, Korea, vol. 206, pp. 129-133, Sept. 22-24, 2011, doi: https://doi.org/10.1007/978-3-642-24106-2_17.
[9] S. H. Oh, "Contour Plots of Objective Functions for Feed-Forward Neural Networks," Int. J. of Contents, vol. 8, no. 4, pp. 30-35, Dec. 2012, doi: https://doi.org/10.5392/IJoC.2012.8.4.030.
[10] S. H. Oh and H. Wakuya, "Comparison of Objective Functions for Feed-Forward Neural Network Classifiers Using Receiver Operating Characteristics Graph," Int. J. of Contents, vol. 10, no. 1, pp. 23-28, Mar. 2014, doi: https://doi.org/10.5392/IJoC.2014.10.1.023.
[11] C. Wang and , Principe, J. C. Princope, "Training Neural Networks with Additive Noise in the Desired Signal," IEEE Trans. Neural Networks, vol. 10, pp. 1511-1517, 1999, doi: https://doi.org/10.1109/72.809097.
[12] K. Liano, "Robust Error Measure for Supervised Neural Network Learning with Outliers," IEEE Trans. Neural Networks, vol. 7, pp. 246-250, Jan. 1996, doi: https://doi.org/10.1109/72.478411.
[13] A. van Ooyen and B. Nienhuis, "Improving the Convergence of the Backpropagation Algorithm," Neural Networks, vol. 4, pp. 465-471, 1992, doi: https://doi.org/10.1016/0893-6080(92)90008-7.
[14] S. H. Oh, "Improving the error back-propagation algorithm with a modified error function," IEEE Trans. Neural Networks, vol. 8, pp. 799-803, May. 1997, doi: https://doi.org/10.1109/72.572117.
[15] J. B. Hampshire and A. H. Waibel, "A Novel Objective Function for Improved Phoneme Recognition Using Time-Delay Networks," IEEE Trans. Neural Networks, vol. 1, pp. 216-218, Jun. 1990, doi: https://doi.org/10.1109/72.80233.
[16] S. Horiguchi, D. Ikami, and K. Aizawa, "Significance of Softmax-Based Features in Comparison to Distance Metric Learning-Based Features," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 42, no. 5, pp. 1279-1285, May. 2020, doi: https://doi.org/10.1109/TPAMI.2019.2911075.
[17] S. H. Oh, "Probabilistic Target Encoding of Neural Networks with Softmax Output Nodes," International Journal of Contents, vol. 18. no. 4, pp. 102-108, Dec. 2022, doi: https://doi.org/10.5392/IJoC.2022.18.4.102.
[18] S. H. Oh, "Statistical Analyses of Cross-Entropy Error Function with Probabilistic Target Encoding for Training Neural Networks," International Journal of Contents, vol. 21. no. 1, pp. 88-92, Mar. 2025, doi: https://doi.org/10.5392/IJoC.2025.21.1.088.
[19] J. W. Shim, "Enhancing cross entropy with a linearly adaptive loss function for optimized classification performance," Scientific Reports, vol. 14, 27405, 2024, doi: https://doi.org/10.1038/s41598-024-78858-6.
[20] Y. Zhou, et al., "MPCE: A maximum probability based cross entropy loss function for neural network classification," IEEE Access vol. 7, pp. 146331-146341, 2019, doi: https://doi.org/10.1109/ACCESS.2019.2946264.
[21] X. Li, et al., "Dual cross-entropy loss for small-sample fine-grained vehicle classification," IEEE Trans. Veh. Technol. vol. 68, pp. 4204-4212, 2019, doi: https://doi.org/10.1109/TVT.2019.2895651.
[22] Y. Ho and S. Wookey, "The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling," IEEE Access, vol. 8, pp. 4806-4813, 2020, doi: https://doi.org/10.1109/ACCESS.2019.2962617.
[23] S. H. Oh, "Error Back-Propagation Algorithm for Classification of Imbalanced Data," Neurocomputing, vol. 74, pp. 1058-1061, 2011, doi: https://doi.org/10.1016/j.neucom.2010.11.024.
[24] S. H. Oh, "Improving the Error Back-Propagation Algorithm for Imbalanced Data Sets," Int. J. of Contents, vol. 8, no. 2, pp. 7-12, Jun. 2012, doi: https://doi.org/10.5392/IJoC.2012.8.2.007.
[25] S. H. Oh, "A statistical perspective of neural networks for imbalanced data problems," Int. J. of Contents, vol. 7, no. 3, pp. 1-5, Sep. 2011, doi: https://doi.org/10.5392/IJoC.2011.7.3.001.
[26] S. H. Oh, "Improving the Water Level Prediction of Multi-Layer Perceptron with a Modified Error Function," Int. J. of Contents, vol. 13, no. 4, pp. 23-28, Dec. 2017, doi: https://doi.org/10.5392/IJoC.2017.13.4.023.
[27] J. J. Hull, "A Database for Handwritten Text Recognition Research," IEEE Trans. Pattern Anal. Machine Intell., vol. 16, no. 5, pp. 550-554, May. 1994, doi: https://doi.org/10.1109/34.291440.