Analysis of Factors Influencing Data-Parallel CNN Models in Image Classification

Hao Yu[1]

[1] College of Robotics, Beijing Union University, Beijing, 100020, China
2019250360064@buu.edu.cn

Abstract. This paper investigates the factors influencing data-parallel convolutional neural network (CNN) models in the context of image classification. This paper analyzes various factors, including batch size, learning rate, loss function, and regularization techniques, to understand their impact on the performance of data-parallel CNN models. By conducting experiments using diverse datasets, such as CIFAR-10, this paper evaluates the effects of these factors on model accuracy, convergence speed, and training efficiency. Based on the research findings, this paper discovered that an appropriate batch size could significantly improve the accuracy and convergence speed of the model. Smaller batch sizes can enhance model sensitivity but may result in slower convergence. Conversely, larger batch sizes can expedite convergence but might lead to overfitting. Additionally, this paper observed that appropriately adjusting the learning rate can further enhance model accuracy and training efficiency. Finally, using regularization techniques, such as L1 or L2 regularization, effectively controls model overfitting, thereby improving its generalization capabilities. These findings provide practical guidance for optimizing data-parallel CNN models and can assist researchers and practitioners in designing more efficient and accurate deep learning systems.

Keywords: data-parallel, CNN, image classification, batch size, learning rate, loss function, regularization techniques, training efficiency.

1 Introduction

Convolutional neural networks (CNNs) have become highly effective tools in computer vision for accomplishing accurate image classification, a fundamental task in this field. However, as the scale of data and model complexity increases, the computational resources of a single GPU may not be sufficient to train large-scale CNN models. To fully harness the parallel computing capabilities of modern computational platforms, data parallelism has become a critical approach for accelerating deep learning training [1].

However, implementing data parallelism involves a range of parameter and method choices, including batch size, learning rate, loss function, and regularization techniques. While several factors greatly influence the performance and training time

P. Kar et al. (eds.), *Proceedings of the 2023 International Conference on Image, Algorithms and Artificial Intelligence (ICIAAI 2023)*, Advances in Computer Science Research 108,
https://doi.org/10.2991/978-94-6463-300-9_100

of CNN models, there are still numerous unanswered questions within this domain. Therefore, a comprehensive analysis of these influencing factors and their interrelationships is essential to optimize data-parallel CNN models for image classification tasks [2].

This paper aims to conduct a thorough analysis of the factors influencing data-parallel CNN models in the context of image classification. By systematically adjusting and comparing different combinations of parameters and methods, such as batch size, learning rate, loss function, and regularization techniques, I explore their effects on model performance and training time, revealing underlying mechanisms and interactions. Such research will contribute to a deeper understanding of optimization strategies for data-parallel methods in image classification, providing guidance and insights for constructing more efficient and accurate image classification models [3].

This paper investigated the impact of various factors on the performance of VGG16, ResNet18, and LeNet models trained and tested on the CIFAR-10 dataset. Specifically, this paper explored the effects of different batch sizes, learning rates, loss functions, and regularization techniques on metrics such as loss, accuracy, and training time. Furthermore, this paper implemented data parallelism for VGG16 and LeNet models and performed data-parallel training and testing on the CIFAR-10 dataset using VGG16. Lastly, this paper attempted to reproduce the Vpipe method [4].

By conducting a comprehensive analysis of the factors influencing data-parallel CNN models in image classification, this paper can provide guidance and insights for constructing more efficient and accurate image classification models [5].

2 Methods

2.1 Dataset

In this study, the CIFAR-10 dataset was chosen as the benchmark dataset. CIFAR-10 is widely used in image classification tasks and consists of 50,000 training images and 10,000 testing images. Each image has a resolution of 32x32 pixels. The dataset is divided into 10 distinct classes, including everyday objects. Please refer to Figure 1 for an illustration of the CIFAR-10 dataset.

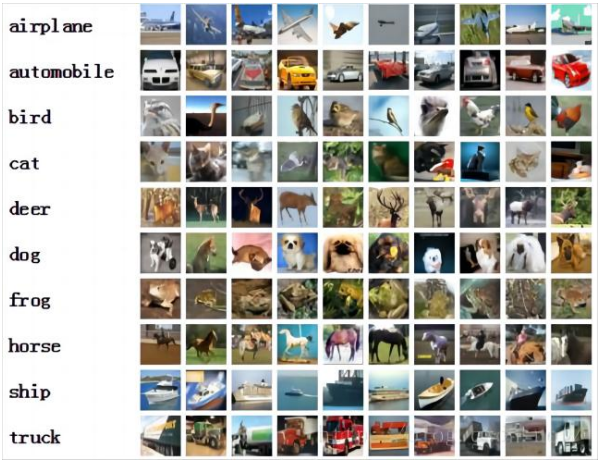

Fig. 1. Legend for CIFAR-10

(Reference: https://blog.csdn.net/weixin_42956785/article/details/89076061)

The CIFAR-10 dataset provides a diverse set of images with various backgrounds, lighting conditions, and object orientations. The presence of similarities and subtle differences between different classes poses challenges for image classification algorithms. The dataset is designed to evaluate the generalization capabilities of machine learning models and is commonly used to assess the performance of CNN architectures.

Researchers and practitioners often use the CIFAR-10 dataset to train and test their image classification models. It is relatively small size and manageable complexity makes it an ideal choice for conducting experiments, exploring different algorithms, and comparing the performance of various models. Each image is labeled with a ground-truth label, enabling supervised learning approaches and facilitating performance evaluation based on accuracy metrics.

The CIFAR-10 dataset has been instrumental in driving progress in the field of computer vision, allowing researchers to develop and assess cutting-edge image classification models. Its widespread usage and availability have fostered the development of innovative techniques and algorithms, contributing to continuous improvements in image classification performance [6].

2.2 Model Architectures

This paper employed three popular CNN architectures: VGG16, ResNet18, and LeNet.

VGG16: A CNN architecture consisting of 16 convolutional layers and 3 fully connected layers. Its main characteristics are a simple hierarchical structure, small convolutional kernel size (3x3), and repeated usage of pooling layers [7].

ResNet18: A deep CNN architecture with skip connections, suitable for training deep networks while maintaining performance. It is relatively lightweight, with 18

layers, striking a balance between model complexity and computational efficiency [8].

LeNet: A pioneering CNN model was developed, incorporating two convolutional layers, pooling layers, and fully connected layers. Its design philosophy involves extracting local features through convolutional layers, reducing spatial dimensions through pooling layers, and performing classification through fully connected layers [9].

2.3 Parallelization Strategy

The focus of this experiment is to utilize data parallelism for training and testing a CNN model. Model parallelism has also been employed during the initial setup.

Data parallelism is a strategy where training data is divided into batches and processed parallel across multiple devices. Model parameters are shared among the devices, and each device computes gradients and updates the parameters using different data batches. This approach leverages parallel computing to speed up training and handle larger datasets.

Model parallelism involves dividing a large model into parts and assigning them to different devices for parallel processing. Each device handles a subset of the model and exchanges intermediate results to perform computations collectively. Model parallelism is functional when the entire model cannot fit into a single device's memory, especially for deep networks with many parameters and complex structures. It optimizes resource utilization while enabling parallel processing of large-scale models.

2.4 Environment

In this experiment, AutoDL was used for GPU rental, and the experimental environment is shown in Table 1.

Table 1. Experimental Environment.

Component	Specification
Image	PyTorch 1.11.0
Python Version	3.8
CUDA Version	11.3
GPU	2 x NVIDIA RTX 3090 (24GB each)
CPU	30 vCPU Intel(R) Xeon(R) Platinum 8358P @ 2.60GHz

3 Experimental results and analysis

To train and test the CIFAR-10 dataset, the models LeNet and VGG16 will be utilized with model parallelism for both and data parallelism specifically for the VGG16 model.

The dataset comprises 50,000 training images and 10,000 test images, all with dimensions of 32x32 pixels. The chosen batch size is 64.

The cross-entropy loss function will be used during training, and the model parameters will be updated using the stochastic gradient descent (SGD) optimizer. The learning rate (lr) is set at 0.001. The training process will run for a fixed number of epochs, specifically 20 epochs. The experimental results are shown in Table 2.

Table 2. Experiment 1 (Batch size:64, Learn rate:0.001, Loss function: Cross entropy loss function, Regularization: None).

	Training Times	Loss	Accuracy
VGG16 (Data Parallel)	554s	0.129	80.96%
VGG16 (Model Parallel)	224s	0.193	81.32%
LeNet (Model Parallel)	112s	0.134	60.24%

On the CIFAR-10 dataset, both Data-Parallel and Model Parallel training methods achieve similar accuracy for the VGG16 model. However, Model Parallel training requires less time. The VGG16 model trained with Model Parallel achieves an accuracy of 81% in a shorter time compared to Data Parallel training.

In contrast, the LeNet model trained with Model Parallel achieves a relatively lower accuracy of 60% on the CIFAR-10 dataset. This indicates that further optimization or considering alternative models may be necessary for improving the performance of the LeNet model on this dataset.

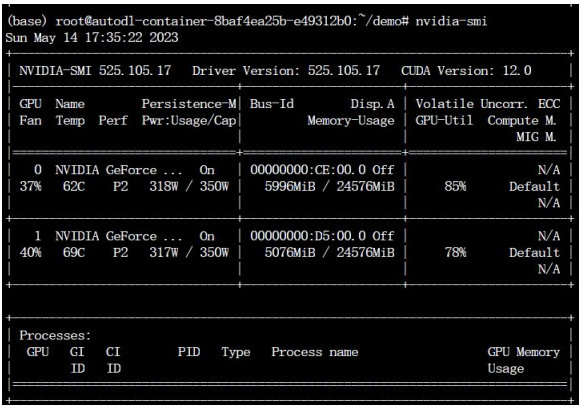

Fig. 2. Experiment 2 (GPU usage rate). (Photo credit: Original)

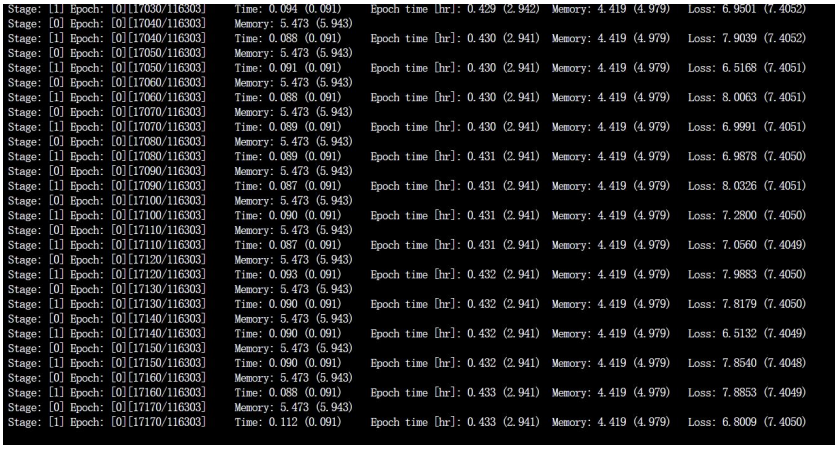

Fig. 3. Experiment 2 (Vpipe execution process). (Photo credit: Original)

Overall, when training on the CIFAR-10 dataset, Model Parallel training method demonstrates an advantage over Data Parallel in terms of time efficiency while achieving similar accuracy. However, it is important to note that the LeNet model performs relatively poorly on this dataset, suggesting the need for further optimization or alternative models. The following is the replication of Vpipe, as shown in Figure 2 and Figure 3.

The data parallel of VGG16, ResNet18, and LeNet models were used to train and test the CIFAR-10 dataset, and the effects of different batch sizes, learning rate (Ir), loss function, and regularization methods on training time, average loss value, and accuracy were analyzed.

The dataset used in this study includes 50,000 training images and 10,000 testing images, all with dimensions of 32x32 pixels. Data augmentation has been applied to the training set. The data augmentation effect is shown in Figure 4.

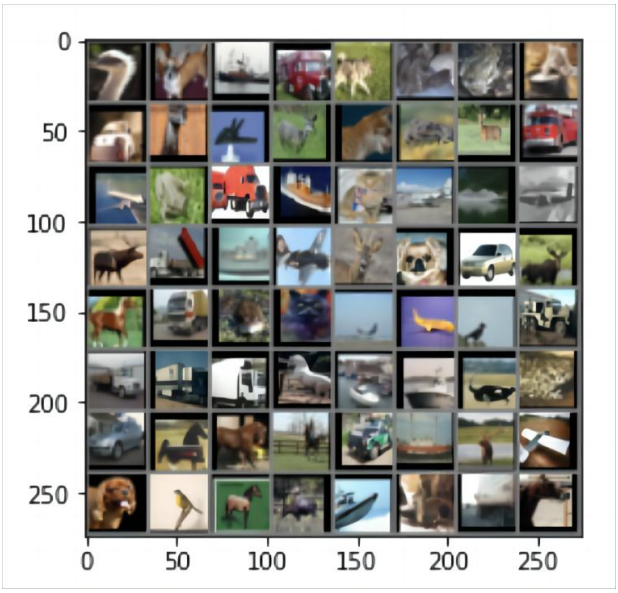

Fig. 4. Data augmentation.

(Photo credit: Original)

Data augmentation is a technique that involves applying various transformation techniques or modifications to expand the training dataset. Its purpose is to increase the size and diversity of the dataset, thereby improving the performance and generalization ability of the model. Data augmentation techniques include image flipping, rotation, scaling and resizing, translation, shearing, zooming, and noise injection [10].

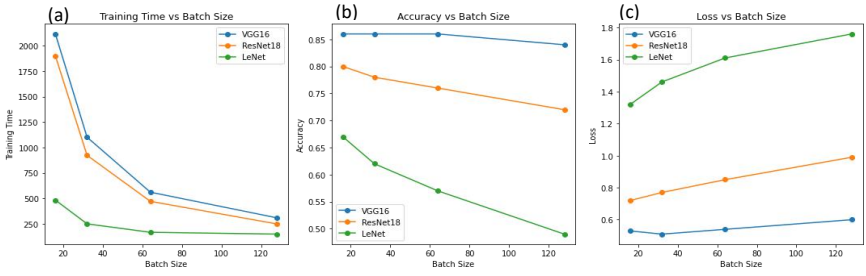

Fig. 5. Experiment 3 different batch sizes (Learn rate:0.001, Loss function: Cross entropy loss function, Regularization: None). (a) Training Time vs Batch Size. (b) Accuracy vs Batch Size. (c) Loss vs Batch Size.

(Photo credit: Original)

The results obtained from Figure 5 with batch sizes set to 16, 32, 64, and 128 are as follows:

According to the experimental results, it is apparent that a larger batch size can shorten the training time. This is mainly attributed to the ability of a larger batch to process more samples in parallel. However, it may have an effect on the accuracy and loss of the LeNet model. In contrast, the VGG16 and ResNet18 models demonstrate more excellent stability when subjected to changes in batch size, with VGG16 generally exhibiting higher accuracy.

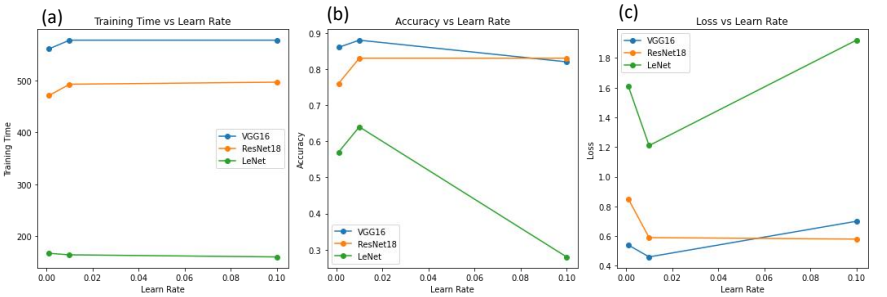

Fig. 6. Experiment 3 different learning rates (Batch size:64, Loss function: Cross entropy loss function, Regularization: None). (a) Training Time vs Learn Rate. (b) Accuracy vs Learn Rate. (c) Loss vs Learn Rate.

(Photo credit: Original)

The results obtained from Figure 6 with learning rate set to 0.001, 0.01, and 0.1 are as follows:

Based on the experimental findings, it is evident that the choice of learning rate substantially impacts the model's performance. A learning rate of 0.01 generally yields good results, enabling the model to achieve the most appropriate levels of accuracy and loss. However, a higher learning rate (0.1) may reduce accuracy and increase loss. This could be attributed to the possibility that the higher learning rate leads to excessively large parameter updates, causing the model to overshoot the optimal solution. It may also introduce instability into the training process.

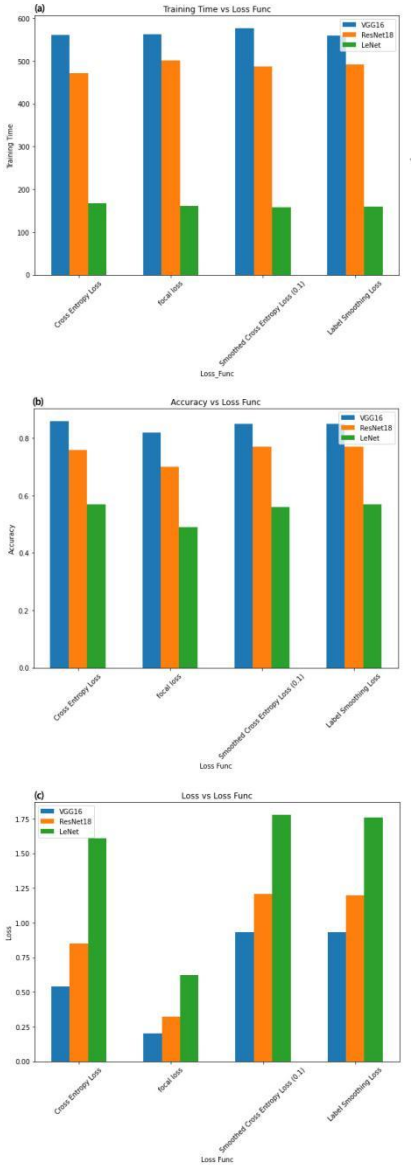

Fig. 7. Experiment 3 different loss function (Batch size:64, Learn rate:0.001, Regularization: None). (a) Training Time vs Loss Func. (b) Accuracy vs Loss Func. (c) Loss vs Loss Func.

(Photo credit: Original)

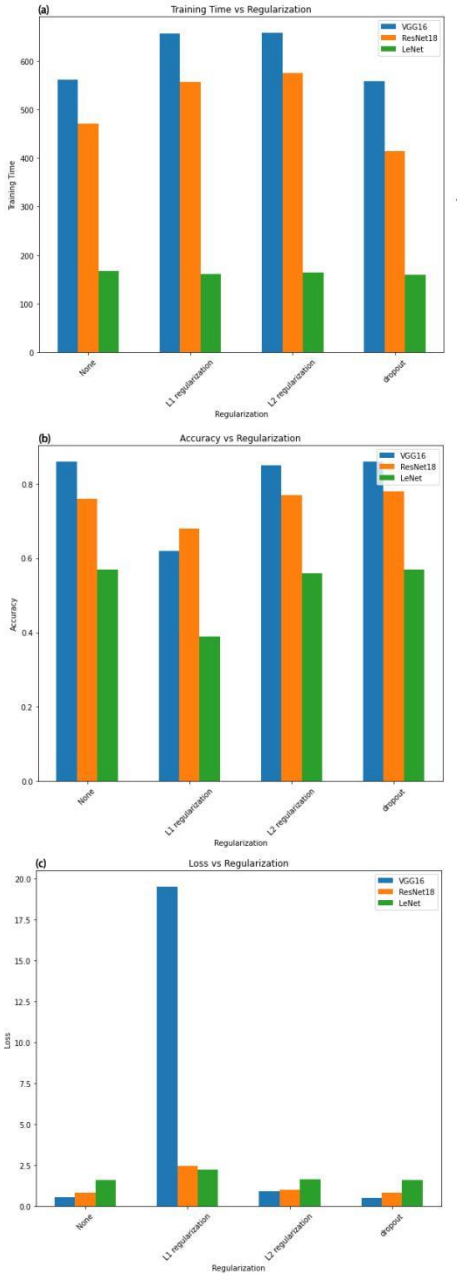

Fig. 8. Experiment 3 different regularizations (Batch size:64, Learn rate:0.001, Loss function: Cross entropy loss function). (a) Training Time vs Regularization. (b) Accuracy vs Regularization. (c) Loss vs Regularization.

(Photo credit: Original)

Cross-entropy loss is widely utilized as a loss function in classification tasks. It quantifies the disparity between the predicted probability distribution and the actual distribution of the target labels [11]. Focal Loss is a modification of Cross Entropy Loss that addresses class imbalance by giving more weight to hard and misclassified examples [12]. Smooth Cross Entropy Loss mitigates overconfidence by smoothing the target labels, encouraging the model to be less certain [13]. Label Smoothing Loss introduces noise to the ground truth labels to prevent overfitting and encourage more generalized learning [14]. These different loss functions provide variations in how the model's predictions are compared to the ground truth labels and address specific challenges in classification tasks, such as class imbalance, overconfidence, and overfitting. The selection of the loss function is contingent upon the dataset's characteristics and the specific demands of the classification problem under consideration.

The results obtained from Figure 7 with the loss function set to Cross Entropy Loss, Focal Loss, Smooth Cross Entropy Loss, and Label Smoothing Loss are as above shows. The experimental results clearly demonstrate that the choice of the loss function has a substantial impact on the accuracy and loss of the LeNet model. When using focal loss, all three models can achieve the lowest loss values, indicating that focal loss is more effective in addressing class imbalance issues. Regularization methods are techniques used to reduce overfitting in models by introducing additional penalty terms in the loss function to constrain the complexity of model parameters. Regularization methods help improve the model's generalization ability and prevent it from overfitting the training data.

L1 Regularization: Encourages parameter sparsity by adding the sum of absolute parameter values as a penalty term [15]. L2 Regularization: Promotes a smoother parameter distribution by adding the squared sum of parameters as a penalty term [16]. Dropout Regularization: Randomly sets neuron outputs to zero during training, enhancing model robustness and generalization [17-20]. The results obtained from Figure 8 with regularization set to None, F1 Regularization, F2 Regularization, and Dropout Regularization are as follows: According to the experimental results, it is evident that the inclusion of L1 and L2 regularization leads to a slight increase in training time. This is likely due to the increased computational complexity associated with regularization techniques. However, the training time for the dropout method remains relatively standard, possibly because it reduces the computational burden by randomly excluding a subset of neurons during parameter updates.

The impact of L1 regularization on accuracy and loss values is also noticeable. This could be attributed to the fact that L1 regularization imposes constraints on model parameters to prevent overfitting. However, excessive restrictions may result in the model being underfitted. The model's performance can vary depending on the dataset and model architecture. Therefore, further experimentation and adjustments may be required to determine its optimal state.

4 Discussion

Due to time and resource limitations, this paper was unable to conduct experiments using different numbers of GPUs or increase the number of training epochs. However, based on our current experiment, we have identified several plans and prospects for future investigations:

Model performance optimization: This paper plans to explore different optimization algorithms, learning rate strategies, and hyperparameter tuning methods to enhance model performance and convergence speed.

Further exploration and implementation of diverse data augmentation and preprocessing techniques can enhance the model's robustness and its ability to generalize to new data.

More complex model architectures: This paper aims to investigate the use of more profound or complex model architectures to enhance model performance further.

Adversarial training and defense strategies: Researching adversarial attacks and defense methods will allow us to understand the robustness of models when faced with adversarial samples.

By addressing these aspects, this paper can gain a deeper understanding of optimizing model performance, improving model robustness, and defending against adversarial attacks, thereby advancing the field of image classification and deep learning.

5 Conclusion

In conclusion, this work focused on applying data parallelism and model parallelism for training CNN models on the CIFAR-10 dataset. This paper found that data parallelism accelerated training times by distributing batches across multiple computing devices. This paper also observed the effectiveness of model parallelism in handling large-scale models.

The research's significance lies in the insights gained regarding parameter tuning and the impact on model performance. Additionally, future work can explore advanced optimization algorithms, data augmentation techniques, and more complex model architectures. Adversarial training and defense strategies can also be studied to enhance model robustness against attacks.

Overall, the findings contribute to the field of image classification and deep learning, and further advancements in these areas will lead to improved model performance and applicability in real-world scenarios with larger datasets and complex models.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional Neural Networks. Communications of the ACM. 60, 84–90 (2017).

2. Shi D, Zurada J, Karwowski W, et al: Batch and data streaming classification models for detecting adverse events and understanding the influencing factors. Engineering Applications of Artificial Intelligence 85:72–84 (2019).
3. Korzhebin TA, Egorov AD: Comparison of combinations of data augmentation methods and transfer learning strategies in image classification used in Convolution Deep Neural Networks. 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus) (2021).
4. Cubaynes HC, Fretwell PT:Whales from space dataset, an annotated satellite image dataset of Whales for Training Machine Learning Models. Scientific Data (2022).
5. Tang H: Image classification based on CNN: Models and modules. 2022 International Conference on Big Data, Information and Computer Network (BDICN) (2022).
6. Lv, X.: CIFAR-10 image classification based on Convolutional Neural Network. Frontiers in Signal Processing. 4, (2020).
7. Sarker, S., Tushar, S.N., Chen, H.: High accuracy Keyway Angle identification using VGG16-based learning method. Journal of Manufacturing Processes. 98, 223–233 (2023).
8. Huang, B., Liu, J., Zhang, Q., Liu, K., Li, K., Liao, X.: Identification and classification of aluminum scrap grades based on the RESNET18 model. Applied Sciences. 12, (2022).
9. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature. 521, 436–444 (2015).
10. Patil, M.S., Chickerur, S.: Study of data and model parallelism in distributed deep learning for diabetic retinopathy classification. Procedia Computer Science. 218, 2253–2263 (2023).
11. Lashgari, E., Liang, D., Maoz, U.: Data augmentation for deep-learning-based electroencephalography. Journal of Neuroscience Methods. 346, (2020).
12. Li, L., Doroslovacki, M., Loew, M.: Approximating the gradient of cross-entropy loss function. IEEE Access. 8, (2020).
13. Yi, J., Tao, J., Tian, Z., Bai, Y., Fan, C.: Focal loss for punctuation prediction. Interspeech 2020. (2020).
14. Bruch, S.: An alternative cross entropy loss for learning-to-rank. Proceedings of the Web Conference 2021. (2021).
15. Mezzini, M.: Empirical study on label Smoothing in Neural Networks. CSRN. (2018).
16. Lan, N., Zhang, F.: Seismic data reconstruction based on smoothed L1/2 regularization. SEG Technical Program Expanded Abstracts 2019. (2019).
17. Poernomo, A., Kang, D.-K.: Biased dropout and Crossmap dropout: Learning towards effective dropout regularization in Convolutional Neural Network. Neural Networks. 104, 60–67 (2018).
18. Zhang, B., Zhou, Z., Cao, W., Qi, X., Xu, C., Wen, W.: A new few-shot learning method of bacterial colony counting based on the edge computing device. Biology. 11, 156 (2022).
19. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift, https://arxiv.org/abs/1502.03167.
20. Lyu, Y., Yang, Z., Liang, H., Zhang, B., Ge, M., Liu, R., Zhang, Z., Yang, H.: Artificial Intelligence-assisted Fatigue Fracture Recognition based on morphing and fully convolutional networks. Fatigue & Fracture of Engineering Materials & Structures. 45, 1690–1702 (2022).