ISSN: 2477-5126
e-ISSN:2548-9356

A Comparative Study of MobileNet Architecture Optimizer for Crowd Prediction

Permana Langgeng Wicaksono Ellwid Putra[1], Muhammad Naufal[2*)], Erwin Yudi Hidayat[3]
[123]Teknik Informatika, Fakultas Ilmu Komputer, Universitas Dian Nuswantoro, Semarang
[123]Jl. Imam Bonjol No.207, Kota Semarang, Jawa Tengah, 50131
email: [1]langgeng86@gmail.com, [2]m.naufal@dsn.dinus.ac.id, [3]erwin@dsn.dinus.ac.id

Abstract *– Artificial intelligence technology has grown quickly in recent years. Convolutional neural network (CNN) technology has also been developed as a result of these developments. However, because convolutional neural networks entail several calculations and the optimization of numerous matrices, their application necessitates the utilization of appropriate technology, such as GPUs or other accelerators. Applying transfer learning techniques is one way to get around this resource barrier. MobileNet is an example of a lightweight convolutional neural network architecture that is appropriate for transfer learning. The objective of the research is to compare the performance of SGD and Adam using the MobileNetv2 convolutional neural network architecture. Model training uses a learning rate of 0.0001, batch size of 32, and binary cross-entropy as the loss function. The training process is carried out for 100 epochs with the application of early stop and patience for 10 epochs. Result of this research is both models using Adam's optimizer and SGD show good capability in crowd classification. However, the model with the SGD optimizer has a slightly superior performance even with less accuracy than model with Adam optimizer. Which is model with Adam has accuracy 96%, while the model with SGD has 95% accuracy. This is because in the graphical results model with the SGD optimizer shows better stability than the model with the Adam optimizer. The loss graph and accuracy graph of the SGD model are more consistent and tend to experience lower fluctuations than the Adam model.*

Keyword **–** adam, crowd prediction, mobilenet, optimizer, sgd, transfer learning

I. INTRODUCTION

Artificial intelligence technology has grown quickly in recent years due to reasons like the advent of the Internet, big data, the Internet of Things (IoT), and powerful processing power [1][2]. Convolutional neural network (CNN) technology, which is beneficial for tasks involving images like classifying and segmenting images, has also been developed as a result of these developments [3][4]. However, because convolutional neural networks entail several calculations and the optimization of numerous matrices, their application necessitates the utilization of appropriate technology, such as GPUs or other accelerators [5][6][7].

Applying transfer learning techniques is one way to get around this resource barrier [8][9][10]. These methods reduce the requirement for significant processing resources by solving new problems using pre-trained neural network models. Because it was previously trained using the ImageNet dataset, which contains approximately 1.2 million photos with 1,000 categories [11], VGG, ResNet, and MobileNet are the example of transfer learning architecture. Among these options, VGG and ResNet stand out for their elevated complexity, owing to their extensive layer structures[12][13]. The primary benefit of utilizing VGG and ResNet lies in their capability to abstract features at a profound level. however, this comes at the cost of increased computational demands[13].

On the other hand, MobileNet represents a more lightweight architectural approach with fewer layers that is designed for mobile and embedded vision applications[14]. Despite its reduced complexity when compared to VGG and ResNet, MobileNet remains adept at delivering valuable feature representations for a diverse array of transfer learning tasks, characterized by a heightened level of efficiency[15][16].

The purpose of this research is to implement a lightweight transfer learning architecture, MobileNet, to compare two optimization methods Adam and SGD. Optimization plays a key role in the training process of neural networks, and various types of optimization algorithms have been developed, including Adam (Adaptive Moment Estimation), SGD (Stochastic Gradient Descent), as well as RMSprop (Root Mean Square Propagation), and Adagrad (Adaptive Gradient).

The objective of the research is to compare the performance of SGD and Adam using the MobileNetv2 convolutional neural network architecture. Optimization plays an important role in the neural network training process, and several types of optimizers have been developed, including Adam (Adaptive Moment Estimation), SGD (Stochastic Gradient Descent), RMSprop (Root Mean Square Propagation), and Adagrad (Adaptive Gradient) [17][18][19]. Adam and SGD were chosen as optimization methods because SGD is effective, interpretable, and simple[20]. While Adam optimizer more complex but can automatically adapt the learning rate for each parameter[21]. Both optimizers have their unique advantages that can be compared in the transfer learning process.

II. RELATED RESEARCH

Optimizer comparisons have been carried out in several previous studies, including research conducted by Poojary et al. In this study, Adam, SGD, and RMSprop optimizers were compared using the Kaggle Cat vs. Dog dataset with InceptionV3 and ResNet50 architectures. The results of this study show that the SGD optimizer has slightly better performance compared to Adam and RMSprop on both architectures used. In addition, the SGD performance graph also shows better stability compared to the other two optimizers on the dataset[22]. Another study was conducted by Solanke et al. The study compared Adam, Adadelta, Adagrad, and RMSprop optimizers on the NSL-KDD dataset. The average results found in this study were Adam with accuracy 0.999, RMSprop with accuracy 0.98, Adagrad with accuracy 0.91, and Adadelta with accuracy 0.93[23]. In addition,

corresponding author :Permana Langgeng Wicaksono Ellwid Putra
Email: langgeng86@gmail.com

research conducted by Lydia et al. also compares optimizers Adagrad, Adadelta, RMSprop, Adam, and SGD on different datasets, such as Coil-100, Caltech-101, and MNIST, using artificial neural networks. The results showed that the Adagrad optimizer achieved excellent accuracy on all three datasets, 0.9999 on Coil-100, 0.9986 on Caltech-101, and 0.9329 on MNIST[24].

The research studies described above compare various types of commonly used optimizers intending to find out which optimizer is most effective for a particular dataset or method. This is because each dataset and technique have unique characteristics, which can produce different comparison results. Based on this, this research compares two optimizers, Adam and SGD, on a crowd prediction dataset using transfer learning.

III. RESEARCH METHOD

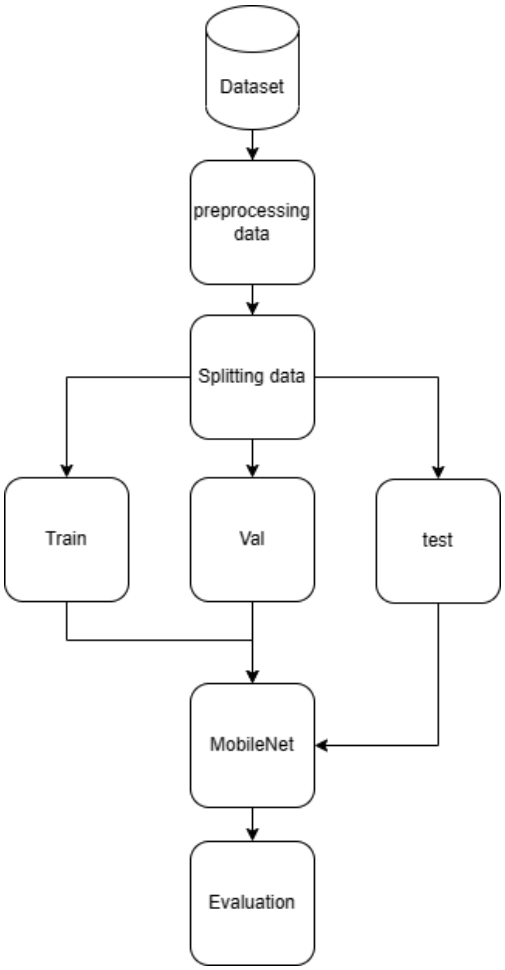

Figure 1. Research Flowchart

The research begins by undertaking the dataset preparation stage, illustrated in Figure 1. Following this, the data from the dataset goes through a preprocessing stage to make it suitable for model training. Once the preprocessing is complete, the data is partitioned into three sets: training, validation, and testing. In the training process, the model learns from the training set, and its performance is assessed using various evaluation metrics, including a loss graph, accuracy graph, confusion matrix, and classification report. These evaluation metrics provide valuable insights into the model's effectiveness and its ability to correctly classify and predict outcomes.

A. Dataset

This research uses the Crowd Human dataset, an opensource accessible collection of crowd photographs in a variety of positions, settings, and sizes [25][26]. This dataset has 15,000 images overall.

B. Dataset Preprocessing

Figure 2 (a) Non-Crowd Original Image (b) Non-Crowd Preprocessing Image Result (c) Crowd Original Image (d) Crowd Preprocessing Image Result

The process for preparing the dataset for model training, as depicted in Figures 2, involves several steps. First, the images in the dataset are cropped to highlight the significant areas of each image and categorized into two classes: Crowd and Non-Crowd. Next, the cropped images are resized to a uniform size of 224 by 224 pixels, which is the input format required for MobileNetV2. Lastly, the pixel intensity data of each image is normalized, converting the values to a range between 0 and 1. This normalization step ensures consistency and better performance during the model training stage.

C. Splitting data

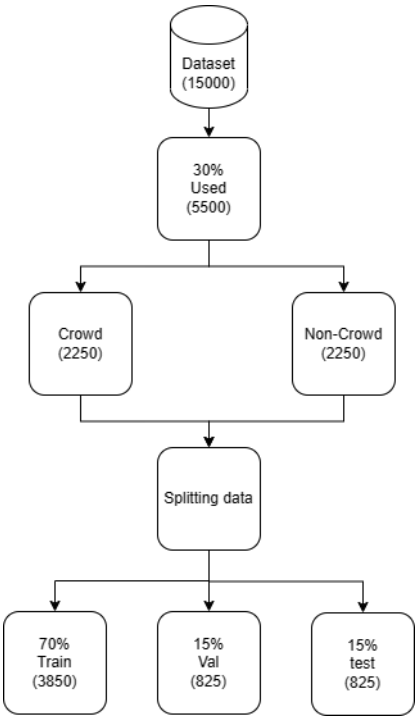

Figure 3 Dataset Splitting

As seen in Figure 3, this study uses only 30% of the entire available dataset, which consists of 5500 images. This dataset is divided into two classes, 2250 images for the crowd class and 2250 images for the non-crowd class. Furthermore, the dataset was divided into three parts, with a proportion of 70% each for training data (3850 images), 15% for validation data (825 images), and 15% for testing data (825 images).

D. Modeling MobileNetv2

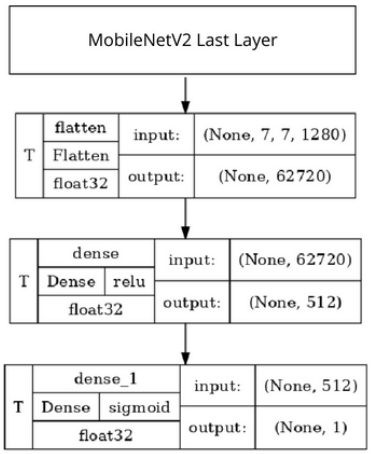

Figure 4 MobileNetv2 Architecture Layers

In this study, the convolutional neural network model used is MobileNetv2. This model has an initial full convolution layer with 32 filters, followed by 19 residual bottleneck layers [27]. These layers will be "frozen", meaning that their weights will not be updated during model training. This is because in the original dataset, ImageNet has been trained with the same knowledge of images of people. Furthermore, three training layers are added underneath as can be seen in Figure 4.

The flatten layer, dense layer, and final dense layer are the added layers as shown in Figure 3. There are 512 and 1 neuron in the dense layer and the final layer, respectively. The ReLu (Rectified Linear Unit) activation function, which aids in overcoming the disappear gradient issue and enhancing training effectiveness, is also used in the dense layer[28]. To avoid overfitting, L2 kernel regulation is also applied to the dense layer. The flatten layer is used to create a one-dimensional vector from the output of the previous layer [29][30]. The final dense layer is to digest data and investigate more intricate patterns from the preceding layer [31][32]. In the final dense sigmoid is used. Sigmoid is an activation function designed for binary classification tasks. Its primary purpose is to transform the model's output into a probability value ranging from 0 to 1[33].

E. Optimizer

The optimizations used in this research are Sigmoid for final layer, Adam (Adaptive Moment Estimation) and SGD (Stochastic Gradient Descent).

1. Adam (Adaptive Moment Estimation)

Adam is a stepwise optimization algorithm on a random objective function that uses adaptive learning rates for each parameter. Adam combines the first and second moments of the gradients to update the parameters[34]. The formula of Adam algorithm shown in equation (1), (2), (3), (4), and (5).

$$m1 = beta1 \times m1 + (1 - beta1) \times gradient \quad (1)$$
$$m2 = beta2 \times m2 + (1 - beta2) \times gradient \quad (2)$$
$$m1_{hat} = m1 \div (1 - beta1^{t}) \quad (3)$$
$$m2_{hat} = m2 \div (1 - beta2^{t}) \quad (4)$$
$$p = p - lr \times m1_{hat} \div (\sqrt{m2_{hat}} + epsilon) \quad (5)$$

Where m is the momentum, beta is the recurrence parameter, gradient is the gradient of the loss function for the parameter, p is the parameter, t is the iteration, lr is the learning rate, and epsilon is a small number to avoid division by zero.

2. SGD (Stochastic Gradient Descent)

SGD is a simple algorithm that updates the parameters in the direction of the negative gradient of the loss function, making it computationally efficient[35]. The formula of SGD algorithm shown equation (6).

$$p = \text{parameter} - \text{learning rate} \times \text{gradient} \quad (6)$$

Where gradient is the gradient of the loss function for the parameter.

F. Evaluation

In this study, an evaluation was conducted on two models using Adam's and SGD optimizer. Evaluation is done by comparing several metrics, such as loss graph and accuracy graph to monitor changes in model performance during training. Confusion matrix to measure the performance of the model in performing classification and classification report to summarize the classification evaluation metrics, including precision, and recall for each target class.

IV. RESULT AND DISCUSSION

A. Implementation

Implementation is done by training the MobileNetV2 model using the provided dataset. The model training uses a learning rate of 0.0001, batch size of 32, and binary cross-entropy as the loss function. The training process is carried out for 100 epochs with the application of early stop and patience for 10 epochs.

The training results show that the model with Adam's optimizer at the 58th epoch, has achieved the best performance on the validation data, and there is no significant improvement afterward. The training was stopped using early stop after reaching 58 epochs out of a total of 100 predefined epochs. Meanwhile, in the model training with the SGD optimizer, the learning process continued until it reached the 100th epoch without an early stop.

B. Evaluation

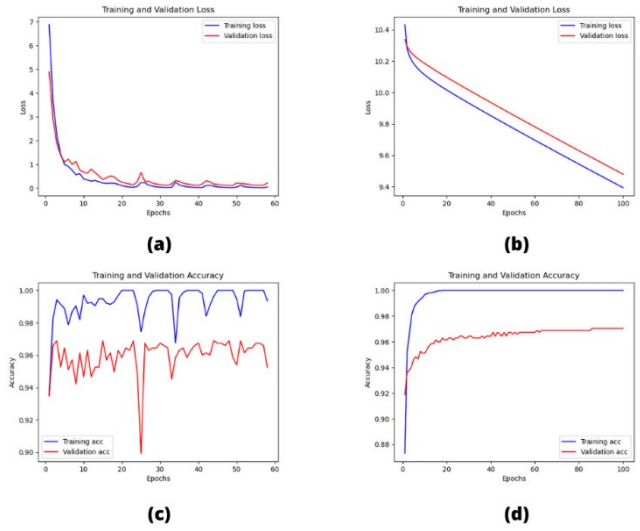

Figure 5 Graph Result (a) Adam Loss (b) SGD Loss (c) Adam Accuracy (d) SGD Accuracy

From Figure 5 (a and b), it can be seen that the loss (train loss and val loss) values in both models decrease with the epoch. However, there is a difference in pattern and stability between the models with Adam's optimizer and SGD. The model with the Adam optimizer initially experiences a sharp decrease in loss value, but then becomes more stable, and the train loss and val loss converge on the graph. This shows that the model got no significant difference between the prediction results on training data and validation data after a few epochs. Meanwhile, the model with the SGD optimizer continues to decrease the loss value until it reaches the last epoch. This indicates that the model with the SGD optimizer has the potential to continue to improve the prediction results on the validation data, but has not yet reached optimal performance at the end of training.

From Figure 5 (c and d), both models have improved accuracy on the training data and validation data. However, there is a difference in stability in the accuracy graph between the models with Adam's optimizer and SGD. The model with Adam's optimizer shows an unstable accuracy graph, fluctuations in accuracy occur in both training data and validation data. This indicates that the model may have difficulty achieving consistency in prediction on both types of data. Meanwhile, the model with the SGD optimizer shows a more stable accuracy graph on both training and validation data. The accuracy graphs on both types of data tend to rise consistently during training. This indicates that the model with the SGD optimizer has more consistent predictions and maintains a stable level of accuracy throughout the training process.

Based on Figure 6, both models have almost the same accuracy, because the confusion matrix results from both show

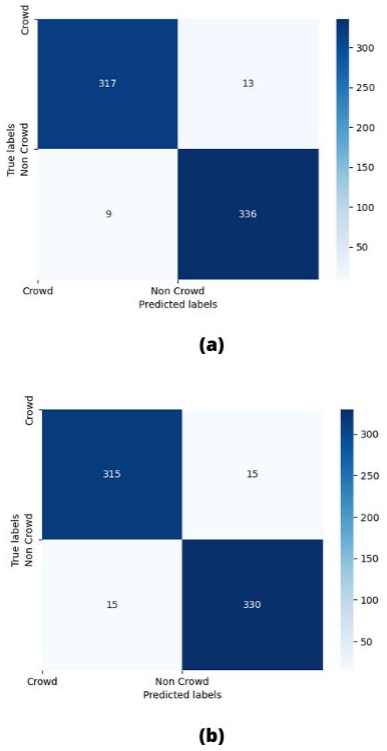

Figure 6. Confusion Matrix (a) Adam (b) SGD

insignificant similarities. The model with Adam's optimizer has a True Positive of 317, False Positive of 9, False Negative of 13, and True Negative of 336. While the model with SGD has a True Positive of 315, False Positive of 15, False Negative of 15, and True Negative of 330. Both models are able to detect target classes (True Positive) and non-target classes (True Negative) well, and have little error in predicting target classes as non-targets (False Negative) and predicting non-target classes as targets (False Positive).

Table 1 Classification Report

Optimizer	Classes	Precision	Recall	Accuracy
Adam	Crowd	0.97	0.96	0.96
	Non-Crowd	0.96	0.97	
SGD	Crowd	0.95	0.95	0.95
	Non-Crowd	0.95	0.95	

Based on Table 1 indicate that both models perform well. The model with Adam has a small advantage in accuracy with 96%, while the model with SGD achieves 95% accuracy.

However, the precision and recall values for both classes show almost the same and high level, close to 0.96. Despite the small difference in accuracy, the evaluation shows that both perform well in predicting the target class and classifying between crowd and non-crowd accurately.

C. Analysis

In the Adam optimizer, the loss graph decreased dramatically at the beginning of training, but then showed instability, and then stopped at epoch 58 out of 100 because the model performance did not change significantly in the last 10 epochs. On the other hand, the SGD optimizer showed a slowly decreasing loss graph, showed better stability in finding the global minimum of the loss function, and the training lasted until it reached epoch 100.

The training graph shows that the model with the SGD algorithm has much higher stability compared to the model using Adam's algorithm, indicating that the SGD model can learn the patterns in the dataset well. Meanwhile, the model with Adam's algorithm faces difficulties in learning patterns from the dataset.

The differences in the loss and accuracy graphs are caused by several factors. First, Adam's optimizer has an adaptive method to adjust the learning rate based on the difference in the calculation of the moments of the gradient which causes a large change in the model parameters at the beginning of training, thus causing instability[36]. Meanwhile, the SGD optimizer that uses a constant learning rate tends to be more stable in finding the global minimum of the loss function[37].

In addition, the crowd dataset has a lot of noise due to the width of the area. SGD optimization is favored because SGD is more resilient to noise in the data as it applies sample-based (stochastic) updates at each step. This helps the model bypass bad local minima and get a better solution overall[38]. On the other hand, Adam's optimization uses exponential momentum and can be more sensitive to noise in the data, which might lead to larger fluctuations in the optimization steps[39].

V. CONCLUSION

Based on the evaluation, both models using Adam's optimizer and SGD show good capability in crowd classification in a simple MobileNetv2 architecture in the given dataset. But the model with SGD optimizer is better than Adam Their performance can be observed through the classification report and confusion matrix. However, the model with the SGD optimizer has a slightly superior performance even with less accuracy than model with Adam optimizer. Which is 95% and 96%. This is because in the graphical results, model with the SGD optimizer shows better stability than the model with the Adam optimizer. The loss graph and accuracy graph of the SGD model are more consistent and tend to experience lower fluctuations than the Adam model.

For further research, the use of the Adam optimizer on noisy dataset can be optimized by making some adjustments to the parameters. For example, the addition of dropout layers and additional dense layers can help reduce overfitting and improve model generalization. In addition, experiments with different learning rates and batch sizes should also be considered to find a better combination for the dataset used. This approach will help in improving the model performance with Adam's optimizer and produce better results in crowd classification tasks.

BIBLIOGRAPHY

[1] W. Hongyuan and D. Mingxing, "OVERVIEW OF THE DEVELOPMENT OF ARTIFICIAL INTELLIGENCE TECHNOLOGY," *Int J Res Eng Technol*, vol. 07, no. 08, pp. 92–95, Aug. 2018, doi: 10.15623/ijret.2018.0708011.

[2] Z. Li, Y. Wang, Y. Ji, and W. Yang, "A survey of the development of artificial intelligence technology," in *Proceedings of 2020 3rd International Conference on Unmanned Systems, ICUS 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020, pp. 1126–1129. doi: 10.1109/ICUS50048.2020.9274952.

[3] Manjunath Jogin, Mohana, M. Madhulika, G. Divya, R. Meghana, and S. Apoorva, "Feature Extraction using Convolution Neural Networks (CNN) and Deep Learning," in *2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, 2018, pp. 2319–2323. doi: 10.1109/RTEICT42901.2018.9012507.

[4] M. Ridley, "Explainable Artificial Intelligence (XAI)," *Information Technology and Libraries*, vol. 41, no. 2, Jun. 2022, doi: 10.6017/ITAL.V41I2.14683.

[5] F. Yan, Z. Zhang, Y. Liu, and J. Liu, "Design of Convolutional Neural Network Processor Based on FPGA Resource Multiplexing Architecture," *Sensors*, vol. 22, no. 16, Aug. 2022, doi: 10.3390/s22165967.

[6] I. Wunderlich, B. Koch, and S. Schönfeld, *An Overview of Arithmetic Adaptations for Inference of Convolutional Neural Networks on Re-configurable Hardware*. 2020.

[7] N. K. Shaydyuk and E. B. John, "FPGA Implementation of MobileNetV2 CNN Model Using Semi-Streaming Architecture for Low Power Inference Applications," in *2020 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*, IEEE, Dec. 2020, pp. 160–167. doi: 10.1109/ISPA-BDCloud-SocialCom-SustainCom51426.2020.00046.

[8] R. Shang, J. Wang, L. Jiao, R. Stolkin, B. Hou, and Y. Li, "SAR Targets Classification Based on Deep Memory Convolution Neural Networks and Transfer Parameters," *IEEE J Sel Top Appl Earth Obs Remote Sens*, vol. 11, no. 8, pp. 2834–2846, Aug. 2018, doi: 10.1109/JSTARS.2018.2836909.

[9] A. W. Salehi *et al.*, "A Study of CNN and Transfer Learning in Medical Imaging: Advantages, Challenges, Future Scope," *Sustainability (Switzerland)*, vol. 15, no. 7. MDPI, Apr. 01, 2023. doi: 10.3390/su15075930.

[10] C. Li, J. Feng, L. Hu, J. Li, and H. Ma, "Review of Image Classification Method Based on Deep Transfer Learning," in *Proceedings - 2020 16th International Conference on Computational Intelligence and Security, CIS 2020*, Institute of Electrical and Electronics Engineers Inc., Nov. 2020, pp. 104–108. doi: 10.1109/CIS52066.2020.00031.

[11] T. Ridnik, E. Ben-Baruch, A. Noy, and L. Zelnik-Manor, "ImageNet-21K Pretraining for the Masses," Apr. 2021, [Online]. Available: http://arxiv.org/abs/2104.10972

[12] K. S. Chandra, A. S. Priya, S. Durga Maheshwari, and D. B. R. Naidu, "Detection of Brain Tumour by integration of VGG-16 and CNN Model," 2020. [Online]. Available: www.ijcrt.org

[13] M. Zawish, S. Davy, and L. Abraham, "Complexity-Driven CNN Compression for Resource-constrained Edge AI," Aug. 2022, [Online]. Available: http://arxiv.org/abs/2208.12816

[14] D. Sinha and M. El-Sharkawy, "Thin MobileNet: An Enhanced MobileNet Architecture," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2019. doi: 10.1109/uemcon47517.2019.8993089.

[15] M. Abu *et al.*, "A Comprehensive Performance Analysis of Transfer Learning Optimization in Visual Field Defect Classification," *Diagnostics*, vol. 12, no. 5, May 2022, doi: 10.3390/diagnostics12051258.

[16] 419~430 I Hepatika Zidny Ilmadina, N. Muhammad, and W. D. Surono, "Drowsiness Detection Based on Yawning Using Modified Pre-trained Model MobileNetV2 and ResNet50," *MATRIK*, vol. 22, no. 3, pp. 419–430, 2023, doi: 10.30812/matrik.v22i3.2785.

[17] J. Kaddour, L. Liu, R. Silva, and M. J. Kusner, "When Do Flat Minima Optimizers Work?," Feb. 2022, [Online]. Available: http://arxiv.org/abs/2202.00661

[18] N. Fatima, "Enhancing Performance of a Deep Neural Network: A Comparative Analysis of Optimization Algorithms," *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal*, vol. 9, no. 2, pp. 79–90, Jun. 2020, doi: 10.14201/adcaij2020927990.

[19] T. Fofana, S. Ouattara, and A. Clement, "Optimal Flame Detection of Fires in Videos Based on Deep Learning and the Use of Various Optimizers," *Open Journal of Applied Sciences*, vol. 11, no. 11, pp. 1240–1255, 2021, doi: 10.4236/ojapps.2021.1111094.

[20] J. Li and Y. Yuan, "Fast Latent Factor Analysis via a Fuzzy PID-Incorporated Stochastic Gradient Descent Algorithm," 2023, Accessed: Sep. 05, 2023. [Online]. Available: https://arxiv.org/ftp/arxiv/papers/2303/2303.03941.pdf

[21] Y. Wang, Z. Xiao, and G. Cao, "A convolutional neural network method based on Adam optimizer with power-exponential learning rate for bearing fault diagnosis," *Journal of Vibroengineering*, vol. 24, no. 4, pp. 666–678, Jun. 2022, doi: 10.21595/jve.2022.22271.

[22] R. Poojary and A. Pai, "Comparative Study of Model Optimization Techniques in Fine-Tuned CNN Models," in *2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA).*, 2019. doi: 10.1109/icecta48151.2019.8959681.

[23] A. V Solanke and G. K. Patnaik, "Intrusion Detection using Deep Learning Approach with Different Optimization," 2020. [Online]. Available: www.ijraset.com

[24] A. A. Lydia and F. S. Francis, "Adagrad - An Optimizer for Stochastic Gradient Descent," *INTERNATIONAL JOURNAL OF INFORMATION AND COMPUTING SCIENCE*, vol. 6, no. 5, pp. 566–568, Sep. 2019.

[25] S. Shao *et al.*, "CrowdHuman: A Benchmark for Detecting Human in a Crowd," Apr. 2018, [Online]. Available: http://arxiv.org/abs/1805.00123

[26] X. Shao *et al.*, "Multi-scale feature pyramid network: A heavily occluded pedestrian detection network based on resnet," *Sensors*, vol. 21, no. 5, pp. 1–15, Mar. 2021, doi: 10.3390/s21051820.

[27] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," Jan. 2018, [Online]. Available: http://arxiv.org/abs/1801.04381

[28] E. Belcore and V. Di Pietra, "LAYING THE FOUNDATION FOR AN ARTIFICIAL NEURAL NETWORK FOR PHOTOGRAMMETRIC RIVERINE BATHYMETRY," in *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, International Society for Photogrammetry and Remote Sensing, Aug. 2022, pp. 51–58. doi: 10.5194/isprs-archives-XLVIII-4-W1-2022-51-2022.

[29] B. Kaddar, H. Fizazi, M. Hernandez-Cabronero, V. Sanchez, and J. Serra-Sagrista, "DivNet: Efficient Convolutional Neural Network via Multilevel Hierarchical Architecture Design," *IEEE Access*, vol. 9, pp. 105892–105901, 2021, doi: 10.1109/ACCESS.2021.3099952.

[30] W. Sun, Q. Wei, L. Ren, J. Dang, and F. F. Yin, "Adaptive respiratory signal prediction using dual multi-layer perceptron neural networks," *Phys Med Biol*, vol. 65, no. 18, Sep. 2020, doi: 10.1088/1361-6560/abb170.

[31] A. M. Javid, S. Das, M. Skoglund, and S. Chatterjee, "A ReLU Dense Layer to Improve the Performance of Neural Networks," Oct. 2020, [Online]. Available: http://arxiv.org/abs/2010.13572

[32] C. Fang, H. He, Q. Long, and W. J. Su, "Exploring Deep Neural Networks via Layer-Peeled Model: Minority Collapse in Imbalanced Training," Jan. 2021, doi: 10.1073/pnas.2103091118.

[33] X. Wang, H. Ren, and A. Wang, "Smish: A Novel Activation Function for Deep Learning Methods," *Electronics (Switzerland)*, vol. 11, no. 4, Feb. 2022, doi: 10.3390/electronics11040540.

[34] W. Pengtao, "Based on adam optimization algorithm: Neural network model for auto steel performance prediction," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Nov. 2020. doi: 10.1088/1742-6596/1653/1/012012.

[35] Institute of Electrical and Electronics Engineers, IEEE Computational Intelligence Society, and Victoria University of Wellington, *2019 IEEE Congress on Evolutionary Computation (CEC) : 2019 proceedings.*

[36] J. Yun, "An Efficient Approach to Mitigate Numerical Instability in Backpropagation for 16-bit Neural Network Training," Jul. 2023. [Online]. Available: http://arxiv.org/abs/2307.16189

[37] L. Wu and C. Ma, "How SGD Selects the Global Minima in Over-parameterized Learning: A Dynamical Stability Perspective," in *32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*, 2018.

[38] F. Kunstner, J. Chen, J. W. Lavington, and M. Schmidt, "Noise Is Not the Main Factor Behind the Gap Between SGD and Adam on Transformers, but Sign Descent Might Be," Apr. 2023, [Online]. Available: http://arxiv.org/abs/2304.13960

[39] L. Hodgkinson and M. W. Mahoney, "Multiplicative noise and heavy tails in stochastic optimization," Jun. 2020, [Online]. Available: http://arxiv.org/abs/2006.06293