A Machine Learning Approach for Face Mask Detection System with AdamW Optimizer

Leong Kah Meng
School of computing
Asia Pacific Univeristy of Technology and Innovation
Kuala Lumput, Malaysia
tp059493@mail.apu.edu.my

Ho Hooi Yi
School of computing
Asia Pacific Univeristy of Technology and Innovation
Kuala Lumput, Malaysia
tp059380@mail.apu.edu.my

Ng Bo Wei
School of computing
Asia Pacific Univeristy of Technology and Innovation
Kuala Lumput, Malaysia
tp072644@mail.apu.edu.my

Lim Jia Xin
School of computing
Asia Pacific Univeristy of Technology and Innovation
Kuala Lumput, Malaysia
tp067426@mail.apu.edu.my

Zailan Arabee Abdul Salam
School of computing
Asia Pacific Univeristy of Technology and Innovation
Kuala Lumput, Malaysia
zailan@mail.apu.edu.my

***Abstract*— As Adam optimizer's learning rate decay hyperparameter has recently been deprecated, this journal article focuses to not only provide an alternate optimizer, but also compare the performance of the said optimizer, AdamW, with the Adam optimizer using a face mask detection model. This study experiments with different weight decay values and finds that a weight decay of 0.00009 with the AdamW optimizer consistently achieves a 98% accuracy rate. Aside from that, this study also discusses the differences between Adam with L2-regularization and AdamW on how the weight decay is decoupled from the Adam optimizer's gradient-base update that impacts the performance of AdamW. Overall, the study provides insights to those new to AdamW and looking for a starting point in optimizing deep learning models.**

Keywords — Convolutional neural network (CNN), deep learning, face mask detection model, AdamW, weight decay, normalized weight decay

I. Introduction

With COVID-19 cases gradually being forgotten by many across the world in 2023, the borders are slowly being reopened for tourists to boost the economic growth in these countries. The thing with humans is that they tend to have a habit of forgetting instead of learning from the past, thus causing history to repeat itself. While COVID-19 cases are slowly subsiding, it will only be a matter of time before a new variant comes in for the next outbreak if a country is not careful. Currently, new COVID-19 variants are still being found with the most recent one being XBB.1.5 that was reported on the 24th of February 2023 (Medicine, 2023).

That said, technologies such as face mask detection systems can help to promote safety and prevent the spread of COVID-19 in public spaces (Rao, Devi, Dileep, & Ram, 2020). These systems implements machine learning algorithms and computer vision to detect whether civilians are wearing masks in public areas such as airports, train stations, and shopping centres. By identifying individuals who are not wearing masks, authorities can take appropriate measures to enforce mask-wearing policies and reduce the risk of transmission. While these systems are not a silver bullet solution, they can be a useful tool for preventing the spread of COVID-19 and protecting public health. In this study, one aims to enhance the face mask detection system's accuracy as well as its consistency by altering its optimizer to a more scale-free optimizer; one that is suitable for deep learning scenarios.

II. Literature Review

With the increasing popularity of deep learning algorithms such as CNN in the development of computer vision systems over the past several decades, the inclusion of face mask detection systems to this list was evident in 2020 due to the outbreak of COVID-19. Rao et al. (2020) had proposed a face mask detection system with an altered version of the CNN algorithm, called M-CNN. As the system the authors were developing turned out to be a relatively small model, M-CNN was simply a scaled-down version of the typical CNN. It is due to these alterations that the system was still able to provide a relatively high accuracy of 91.21% given only 1500 sample images to train and test the model with. Aside from that, the authors had also implemented other algorithms such as Scale Invariant Feature Transform (SIFT), Histogram of Oriented Gradients (HOG), and Support Vector Machine (SVM) to extract as well as detect objects and value mappings (Rao, Devi, Dileep, & Ram, 2020). Utilizing all these algorithms had allowed the authors to achieve a computer vision system that is able to perform image classification under harsh lighting conditions as well as various dimensions.

In addition to algorithm selection, noise in training sample images has been found to impact the accuracy of image classification models. Lau et al. (2021) investigated the impact of noise on a Keras Simple CNN model and found that the accuracy rate decreased when noise was present in the training sample images. Using Kaggle as the development platform, the model was trained with six (6) different sets of sample data include a noise-free set of sample images and using the same set of images with different levels of noise ranging from 10 to 50 to generate the other five (5) sets. The training and testing were done in two (2) separate scenarios with the first one being to train and test using sample images in their respective noise levels, and the second scenario training the model with noisy sample images, but is tested using the noise-free sample images. The authors suggest that training sample images should be tied to the purpose of the model, train the model with noisy images if the model was supposed to classify noisy images and vice versa (Lau, Sim, Chew, Ng, & Abdul Salam, 2021). These findings have implications for developing face mask detection systems that are robust to noisy image data.

Upon further research on CNN, it was found that The Decoupled Weight Decay Regularization (DWDR) method was first tokened in this journal article where the authors had implemented the weight decay regularization in the optimizer to enhance the model's accuracy and convergence speed (Loshchilov & Hutter, Decoupled Weight Decay Regularization, 2019). Decoupling the weight decay value from the optimizer's gradient-base rule and applying it directly onto the weight value itself was found to be reason behind these improvements when comparing with Adam with L2-regularization and SGD with momentum. It was also mentioned that the AdamW optimizer was proven to be effective in preventing the model from overfitting as separating the weight decay from the other hyperparameters may improve the optimizer's generalization, especially in deep neural network (DNN) (Loshchilov & Hutter, Decoupled Weight Decay Regularization, 2019). Through this study, the use of AdamW optimizer may allow one in developing accurate and efficient face mask detection systems using CNN.

As for improvements that can be made to streamline the training process of the system, we can consider using transfer learning models such as Big Transfer (BiT) as a general visual representation to be transferred to our downstream task of a face mask detection system. Through a study, it was found that BiT, a DNN that utilizes the transfer learning method that outperforms previous state-of-the-art methods such as ResNet-v2, FixRes, and SwaAV on several benchmark datasets (Kolesnikov, et al., 2020) is computationally efficient and would allow us to save on resources and time if we were to use it following the concept of transfer learning.

III. MATERIALS

A. Dataset

The dataset used in this study consists of a total of 4,000 sample images, with 2,000 images each for masked and unmasked faces. According to the original author of the Face Mask Detection (Maskd) system, the images were collected from 3 different sources – Kaggle datasets, RMFD dataset, and Google Dataset Search, before being split into training and testing sets with a ratio of 8:2.

B. Implementation

The model was implemented using Python 3.8 , TensorFlow 2.11.0, TensorFlow Addons 0.19.0 and NumPy 1.22.4 for tasks ranging from preprocessing, model architecture building, and model training. Additionally, pandas 1.3.5 and Matplotlib were used in data analysis to graph training and validation loss as well as accuracy. The code was run on Google Colab's Python 3 Google Compute Engine that comes with an Intel® Xeon® CPU @ 2.20 Ghz including two cores, a total of 12.7 GB of system RAM and 107.7 GB of disk space with no GPU.

IV. METHODS

A. Preprocessing

All images in the input dataset of face images were resized to a fixed size of 96 x 96 pixels to ensure consistency in size. To increase the size of the training dataset and prevent overfitting, data augmentation techniques such as random rotation, zoom, horizontal flip, and vertical flip were applied to the pre-processed images. Specifically, the images were randomly rotated up to 20 degrees, zoomed up to 15%, horizontally flipped with a probability of 50%, and vertically flipped with a probability of 50%.

B. Model Architecture

The input image is first fed into a 3x3 convolutional layer with 16 filters, followed by max pooling with a 2x2 kernel size. This is repeated with increasing numbers of filters for the next 4 convolutional layers, each followed by max pooling with a 2x2 kernel size. The output of the final convolutional layer is flattened and fed into two fully connected layers with 1024 and 64 nodes respectively. The final output layer has 2 nodes, representing the binary classification of masked faces to unmasked faces from the sample images using the softmax activation function.

C. Model Training

The model was trained using the AdamW optimizer with a learning rate of 0.0005 and a batch size of 32. The training was carried out across 100 epochs and uses a binary cross-entropy loss function. The learning rate of 0.0005 was chosen based on a preliminary hyperparameter search which we found to produce good results and weight decay was added to help prevent overfitting and improve generalization.

D. Evaluation

The trained model was evaluated on the testing set to measure its performance at the end with a classification report, showing the model's precision, recall, and f1-score.

V. ALGORITHM

The algorithm used in the face mask detection system is CNN, a type of neural network that is found to be particularly well-suited when it comes to analyzing two-dimensional data such as images, whether it may be for detection, classification, segmentation, or any other image processing purposes (Ng, Chong, Mohammed, How, & Abdul Salam, 2023). Compared to traditional neural networks such as feedforward neural networks, every neuron in the adjacent layer is found connected to one another and is typically referred to as a dense or fully connected layer. That said, while traditional neural networks are suited for various types of input, it can be computationally expensive. However, in CNN, the input image is said to have passed through a series of convolutional and pooling layers, which allows it to identify and extract features such as edges, corners, and textures (Yamashita, Nishio, Do, & Togashi, 2018). The output from these layers is then passed through one or more fully connected layers, which use weights to combine the features and produce the final output using an activation function. The Adam optimizer is a commonly used optimization algorithm for training CNNs, as it adapts the learning rate based on the history of gradients and adjusts the model parameters more efficiently than traditional optimization algorithms (Bock & Weiß, 2019). The specific architecture and hyperparameters of a CNN will largely depend on the particular task at hand, whether it may be image classification or object detection.

A. Purpose

As the learning rate decay hyperparameter has been recently deprecated and no longer in use for in the latest Keras optimizer, this study was done to find an alternative while also improving the original face mask detection model's performance. Upon further research, it was found that not only was AdamW optimizer a viable option, but also turned out to be an improvement according to pass studies (Loshchilov & Hutter, Decoupled Weight Decay Regularization, 2019) AdamW optimizer was found to provide lower loss rates and better generalization in deep learning scenarios due to the adaptive gradient scaling by decoupling the weight decay from the Adam optimizer's update rule.

B. Parameters

TABLE I. PARAMETERS

Parameter	Value
Learning rate	0.0005
Batch size	32
Number of epochs	100
Loss function	Binary cross-entropy
Optimizer	AdamW
Weight decay	0.0001

VI. DISCUSSION ON IMPLEMENTATION

To start, we will first discuss the parameters shown in TABLE I. In order for a model to converge on an optimum solution in a reasonable amount of time during training, a delicate balance of learning rate, batch size, and number of epochs is a basic requirement. The model might overshoot the optimal solution if the learning rate is set too high and conversely may be stuck in a local optimum if set too low (Wanjau, Wambugu, & Oirere, 2021).

In regard to batch size, a lower batch size would lead to a better optimization of the model, but the training speed of the model will slow down as a result (Jun & Wengdong, 2019). Therefore, choosing the right batch size is essentially a tradeoff between optimization and opportunity cost. On a surface level, one might think that the change in number of epochs has similar effects as batch size. However, the increase in the number of epochs can also cause overfitting while decreasing batch size only has a longer training time as a downside (Saahil & Smitha, 2020). Ultimately, the optimal number of epochs depends on the size and complexity of the dataset and the model's architecture.

As for loss function, it was found that the suitable loss functions for tasks with two outputs consist of binary cross-entropy and hinge loss, where hinge loss is typically used in support vector machines (SVMs) instead, while binary cross-entropy is designed for neural networks (Wang, Ma, Zhao, & Tian, 2022).

In the context of AdamW, weight decay is a hyperparameter that can significantly impact the performance of a model as it controls the amount of regularization applied to the weights during training (Loshchilov & Hutter, Fixing Weight Decay Regularization in Adams, 2018) which can help prevent overfitting and allow for a model with improved generalization to be made. Compared to other hyperparameters in AdamW which influences regularization such as learning rate, weight decay can be seen to have a more direct impact (Loshchilov & Hutter, Decoupled Weight Decay Regularization, 2019). It is due to this very reason we have chosen to modify the weight decay in the AdamW optimizer in our paper to gather findings on the effects of weight decay on a CNN model such as this face mask detection system.

$$g_t = \nabla f_t(w_{t-1}) + \lambda w_{t-1} \tag{1}$$

$$w_t = w_{t-1} - \alpha(\hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)) \tag{2}$$

Formula (1) represents the gradient calculation for the Adam with L2-regularization, which will later be used indirectly in formula (2)'s $\boldsymbol{\hat{m}_t}$ and $\boldsymbol{\hat{v}_t}$ values to obtain the said optimizer's weight value (Loshchilov & Hutter, Decoupled Weight Decay Regularization, 2019).

$$g_t = \nabla f_t(w_{t-1}) \tag{3}$$

$$w_t = w_{t-1} - \alpha(\hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon) + \lambda w_{t-1}) \tag{4}$$

Formula (3) and formula (4) show the gradient formula as well as the weight calculation respectively for the AdamW optimizer based on the study by (Loshchilov & Hutter, Decoupled Weight Decay Regularization, 2019).

Through these formulas, $\boldsymbol{g}$ is the representation for the gradient value and $\boldsymbol{\nabla f_t}$ is the calculation function for g. $\boldsymbol{w_t}$ on the other hand represents the calculated weight value after applying the weight update rule, while $\boldsymbol{w_{t-1}}$ refers to the weight value before the said rule application. As for $\boldsymbol{\alpha}$, it simply refers to the learning rate value. According to (Kingma & Ba, 2015), $\boldsymbol{\hat{m}_t}$ represents the corrected biased of the moving average of gradient, $\boldsymbol{m_t}$. m_t is the mean of the gradient and requires its own value to be included upon calculating m_t itself. As m_t is initially set to 0, this would cause biasness to occur, thus required to be corrected. $\boldsymbol{\hat{v}_t}$ works similarly to $\boldsymbol{\hat{m}_t}$ but instead, v_t represents the uncentered variance of the gradient, or squared gradient. As for epsilon, $\boldsymbol{\varepsilon}$, it is a parameter used in the update rule to ensure that the $\boldsymbol{\hat{m}_t}$is never divided by 0. $\boldsymbol{\lambda}$ is the previously discussed weight decay value. Table II shows the symbols and descriptions for all four (4) of the formulas above.

TABLE II. UPDATE RULE SYMBOLS

Symbol	Description
g	Gradient
∇f	Gradient function
w	Weight
α	Learning rate
$\hat{m}$	Corrected bias of moving average of gradient
$\hat{v}$	Corrected bias of squared gradient
ε	Epsilon
λ	Weight decay

As one may see in formula (1) and formula (2), the weight decay value, λ, was used as part of the gradient's calculation before being indirectly used by $\boldsymbol{\hat{m}_t}$ and $\boldsymbol{\hat{v}_t}$'s calculations respectively. Thus, causing the significance of λ to be lessened. On the other hand, notice how the λ has been decoupled from the gradient formula in formula (3) and instead being included in weight update formula, formula (4). This allows λ to provide a more significant impact to the weight update, making the penalization more effective.

$$\lambda = \lambda_{norms}\sqrt{b/BT} \quad (5)$$

According to the authors of AdamW, it was found that the optimal weight decay value for a model can be determined using the formula as given above, where $\boldsymbol{\lambda_{norms}}$ represents the normalized weight decay value while ***b*** as the batch size. ***B*** on the other hand represents the total training points, and ***T*** for the total epochs (Loshchilov & Hutter, Decoupled Weight Decay Regularization, 2019). As (Loshchilov & Hutter, Decoupled Weight Decay Regularization, 2019), had used a normalized weight decay ranging from 0.025 to 0.05, it was assumed by (Lo, 2021), that this can be a decent starting point. That said, as our model is a relatively small compared to the one used by (Loshchilov & Hutter, Decoupled Weight Decay Regularization, 2019), the normalized weight decay value that was tested for our model ranges from 0.001 to 0.01. Given that the face mask detection system model has a batch size of 32, trained with 3200 sample images, and had a total number of 100 epochs, a good start for the weight decay value, λ, can be calculated to range from 0.00001 to 0.0001.

VII. RESULTS

TABLE III. FACE MASK DETETCION MODEL OUTPUT RESULTS

Weight decay, λ	Run	Accuracy, %
0.0001	1	97
	2	97
0.00001	1	97
	2	98
0.00005	1	97
	2	98
0.00009	1	98
	2	98
	3	98
	4	98

The face mask detection model was trained separately using four (4) different weight decay, λ, values including 0.0001, 0.0001, 0.00005 and 0.00009 as shown in Table III, with at least two runs for each weight decay value.

Throughout the training and testing process of the face mask detection model, λ = 0.0001 was chosen as the initial tested value for the weight decay hyperparameter. It was found that the accuracies for both different runs were 97%, showing a promising output due to its consistency. As our model is considered a simpler model, λ = 0.00001 was carried out and obtained accuracy rates of 97% for the first run, and 98% for the second run; improved accuracy with less consistency. Upon adjusting to a larger λ value, 0.00005, it was again found that the outcomes were similar to the ones from λ = 0.00001. While the accuracy rates for both λ = 0.00001 and λ = 0.00005 can go up to 98%, it was found to be less consistent compared to the models with λ = 0.0001. For the final tested λ, the value of 0.00009 was chosen as it is closer to the first tested weight decay value, λ = 0.0001. The tests had shown promising results of 98% accuracy for both the separate models, making it worth the further validations on these improved accuracies to ensure consistency. Two more separate models were then trained again using the same λ value of 0.00009 to validate the previous two results, and to one's surprise, the accuracy remained at 98%. Not only does this proves that λ = 0.00009 can achieve a higher accuracy of 98% compared to λ = 0.0001 but is also considered to be a significant finding as the consistency is proven to be there.

VIII. CONCLUSION

All in all, the key takeaway point is that AdamW optimizer was found to be a better optimizer compared to Adam with L2-regularization optimizer when it comes to updating and controlling the weight of the model as discussed earlier. Through multiple experiments with different weight decay values, a model with a consistent accuracy of 98% was achieved using the AdamW optimizer with a weight decay of 0.00009. AdamW optimizers are considered to provide state-of-the-art performance when it comes to tasks such as image classification, object detection, segmentation, and other deep learning image processes. As such, models that require high accuracy and convergence speed while working on a rather complex dataset may consider the AdamW optimizer as a viable option. This includes models from the medical field such as medical diagnosis models and automotive industries with their gaining popularity of autonomous driving (Lima, Brito, Martins, Lima, & Pedrosa, 2019; Fujiyoshi, Hirakawa, & Yamashita, 2019). As the accuracy produced by different optimizers may vary from one another, this journal article will provide insights to those who are new to AdamW and are looking for a starting point. To obtain a starting weight decay value, one may use formula (5) to calculate the optimal weight decay range based on their specific models, or by simply implementing the transfer learning method using this study's system.

IX. LIMITATION AND FUTURE ENHANCEMENTS

The one limitation to the AdamW optimizer is the computational power that is considered to be relatively expensive as every weight in the optimizer requires more calculation as well as memory compared to Adam. Not to mention new hyperparameters such as weight decay coefficient in AdamW that was never in the Adam optimizer which also contributes to the increment in computational power required. There are also several unexplored areas when considering AdamW's improvement in performance including the use of batch normalization and dropout techniques. It was found that batch normalization allows the input of every layer to be normalized by controlling the mean and variance of the input to maintain better consistency upon each batch's input (Ioffe & Szegedy, 2015). That said, studies mentioned that there are cases where removing batch normalization from a model may improve the model's accuracy rather than decreasing it, which would be worth the study (Brock, De, Smith, & Simonyan, 2021). The dropout technique on the other hand was said to simply drop random neurons upon passing through each layer to prevent overfitting to occur (Nitish Srivastava, 2014). Finally, it is possible to implement more data augmentation methods such as CutMix, MixUp, gaussian noise, and color jittering to increase the data variation for a more robust model (Tufail, et al., 2022; Hao, et al., 2020; Walawalkar, Shen, Liu, & Savvides, 2021).

ACKNOWLEDGEMENT

We would like to express our sincerest gratitude to Mr. Zailan Arabee Abdul Salam for providing us with insightful feedback based on his expertise throughout the entire research process. What he had provided for us was invaluable and

remained to be an integral instrument in shaping our study paper into what it is today.

REFERENCES

Bock, S., & Weiß, M. (2019). A Proof of Local Convergence for the Adam Optimizer. IEEE Xplore, 1-8.

Brock, A., De, S., Smith, S. L., & Simonyan, K. (2021). High-Performance Large-Scale Image Recognition Without Normalization. Proceedings of the 38th International Conference on Machine Learning (ICML 2021) (pp. 3461-3471). Proceedings of Machine Learning Research (PMLR).

Fujiyoshi, H., Hirakawa, T., & Yamashita, T. (2019). Deep learning-based image recognition for autonomous driving. IATSS Research, 244-252.

Hao, X., Jia, J., Mateen Khattak, A., Zhang, L., Guo, X., Gao, W., & Wang, M. (2020). Growing period classification of Gynura bicolor DC using GL-CNN. Computers and Electronics in Agriculture, 1-12.

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Proceedings of the 32nd International Conference on Machine Learning (pp. 448-456). Proceedings of Machine Learning Research (PMLR).

Jun, H., & Wengdong, Z. (2019). An Adaptive Optimization Algorithm Based on Hybrid Power and Multidimensional Update Strategy. IEEE Access, 19355-19369.

Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations (ICLR) 2015. San Diego, California, USA: Conference Publishing Services (CPS).

Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., & Houlsby, N. (2020). Big Transfer (BiT): General Visual Representation Learning. Conference on Neural Information Processing Systems(34), 11368-11378.

Laarhoven, T. v. (2017). L2 Regularization versus Batch and Weight Normalization. International Conference on Learning Representations (ICLR) 2019. New Orleans, Louisiana, USA: OpenReview.

Lau, Y., Sim, W., Chew, K., Ng, Y., & Abdul Salam, Z. A. (2021). Understanding How Noise Affects The Accuracy of CNN Image Classification. Journal of Applied Technology and Innovation, 23-27.

Lima, T. A., Brito, E. C., Martins, R., Lima, S. G., & Pedrosa, R. P. (2019). Obstructive sleep apnea and quality of life in elderly patients with a pacemaker. J Bras Pneumol.

Lo, A. (2021, December 18). Weight Decay and Its Peculiar Effects. Retrieved from Towards Data Science: https://towardsdatascience.com/weight-decay-and-its-peculiar-effects-66e0aee3e7b8

Loshchilov, I., & Hutter, F. (2018). Fixing Weight Decay Regularization in Adams. ICLR 2018 Conference, 1-13.

Loshchilov, I., & Hutter, F. (2019). Decoupled Weight Decay Regularization. Proceedings of the Sixth International Conference on Learning Representations, 1-18.

Medicine, N. (2023, February 24). What COVID-19 variants are going around in February 2023? Retrieved from Nebraska Medicine: https://www.nebraskamed.com/COVID/what-covid-19-variants-are-going-around

Ng, Y. R., Chong, Y. K., Mohammed, O., How, Y. H., & Abdul Salam, Z. A. (2023). Convolutional Neural Network for Fruit Image. Journal of Applied Technology and Innovation, 21-27.

Nitish Srivastava, G. H. (2014). Dropout: A Simple Way to Prevent Neural Networks from. Journal of Machine Learning Research (JMLR), 1929-1958.

Rao, T. S., Devi, S. A., Dileep, P., & Ram, M. S. (2020). A Novel Approach To Detect Face Mask To Control Covid Using Deep Learning. European Journal of Molecular & Clinical Medicine, 11.

Saahil, A., & Smitha, R. (2020). Significance Of Epochs On Training A Neural Network. International Journal of Scientific & Technology Research, 485-488.

Tufail, A. B., Ullah, K., Ali Khan, R., Shakir, M., Abbas Khan, M., Ullah, I., . . . Ali, M. S. (2022). On Improved 3D-CNN-Based Binary and Multiclass Classification of Alzheimer’s Disease Using Neuroimaging Modalities and Data Augmentation Methods. Journal of Healthcare Engineering.

Walawalkar, D., Shen, Z., Liu, Z., & Savvides, M. (2021). Attentive CutMix: An Enhanced Data Augmentation Approach for Deep Learning Based Image Classification. 2021 International Conference on Computer Vision and Image Processing (CVIP). Springer.

Wang, Q., Ma, Y., Zhao, K., & Tian, Y. (2022). A Comprehensive Survey of Loss Functions in Machine Learning. Annals of Data Science, 187-212.

Wanjau, S., Wambugu, G., & Oirere, A. (2021). Empirical Evaluation of Adaptive Optimization on the Generalization Performance of Convolutional Neural Networks. Kabarak University International Conference On Computing And Information Systems. Nakuru, Kenya.

Yamashita, R., Nishio, M., Do, R. K., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. SpringerOpen, 611–629.