

CRITICAL ANALYSIS OF CNN ARCHITECTURE DEVELOPED FOR COURSEWORK 2

Student Name: Bakhtiyor Sohbnazarov

Student ID: Z22590018

Module: Machine Learning

Updated: 4th of December 2025

Contents

Justification.....	3
Analysis. Critical Thinking.....	3
Strengths and Limitations.....	3
Conclusion	4
References	5

Justification

CIFAR-10 is a simple dataset containing 60,000 32×32 images across one-hot encoded classes. It's lightweight and balanced, making it suitable for testing and developing beginner-friendly CNNs.[1],[4]

In contrast, the Intel Image Classification dataset [2] used earlier in this research introduced a lot more complexity during both architecture design and training, especially since CNN training is more expensive than many other algorithms.

The custom CNN developed for this coursework uses three convolutional blocks with gradually increasing filter sizes and batch normalization [3], [5] to stabilize and speed up training. GlobalAveragePooling is used to reduce parameters while keeping enough spatial information for efficient learning.

Analysis. Critical Thinking

In the final training run, the model reached 68% validation accuracy with over 300,000 parameters. The accuracy is lower than expected, but the model can still be improved with careful tuning and longer training. As mentioned earlier, training takes time and patience because it's computationally expensive.

Hardware also played a major role in this project. Windows machines are widely used today, but for machine-learning work, Linux systems tend to offer better tools, faster debugging, and quicker training. During this coursework, TensorFlow [6],[7] revealed that from version 2.10 onwards, Google stopped developing CUDA support for Windows, which forced me to train on CPU—far slower than GPU training. GPU training on windows still could be applied but it would cause far more complicated problem of versioning dependencies which would affect portability [11]

The learning rate stayed stable during the last run, as shown in the recording, but the validation metrics fluctuated heavily. (Figure 1)

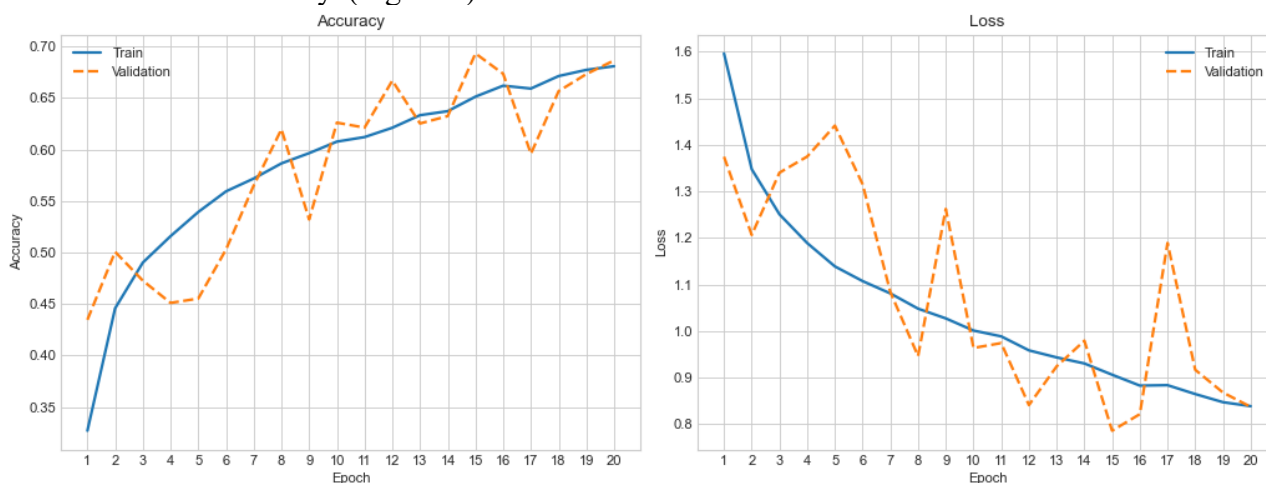


Figure 1. Training Performance

I suspect this was caused by data augmentation, because augmentation constantly changes the images' color and shape, forcing the model to adjust to new patterns each time. For future trainings augmentation parameters should be monitored and possibly altered to smoothly alter images on the fly

Strengths and Limitations

The architecture is simple, yet it shows promising results that can be improved over time with more training, better hardware, and tuned hyperparameters. Because models like this are straightforward to build, anyone with some technical knowledge of machine learning can train their own CNNs for everyday use.

However, CNNs still struggle with classes that look very similar, like cats and dogs, which I observed during training. (Figure 2).

This shows that while CNNs can classify images well, they still need careful training and enough variation in the data to tell very similar objects apart.

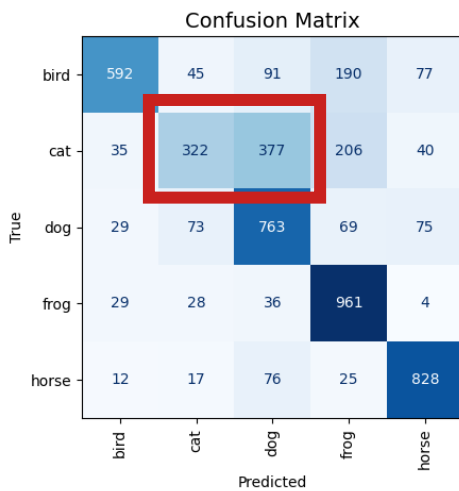


Figure 2. Confusion Matrix of Validation Data [9]

Conclusion

Messy or uneven datasets can significantly affect training, testing, and real-world performance. At the same time, the hardware used for training is equally important, as it directly impacts the speed of training and the efficiency of prototyping models. For this model we trained in this coursework, 68% accuracy is acceptable to use, however future tuning should be considered to achieve better results.

References

- [1] F. Sayah, “CIFAR-10 images classification using CNNs-88%,” Kaggle, 2022. [Online]. Available: <https://www.kaggle.com/code/faressayah/cifar-10-images-classification-using-cnns-88>
- [2] P. K. Bharti, “Intel Image Classification,” Kaggle, 2019. [Online]. Available: <https://www.kaggle.com/datasets/puneet6060/intel-image-classification>
- [3] A. Ghosh, “Batch Norm Explained Visually — How it works and why neural networks need it,” Towards Data Science, Sep. 20, 2021. [Online]. Available: <https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739/>
- [4] A. Querretamontoro, “The best CNN for CIFAR10 from scratch: 93% accuracy,” Medium, Mar. 8, 2024. [Online]. Available: <https://medium.com/@anderaquerretamontoro/the-best-cnn-for-cifar10-from-scratch-93-accuracy-bde35e17fca6>
- [5] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” arXiv, Mar. 2, 2015. [Online]. Available: <https://arxiv.org/abs/1502.03167>
- [6] “Convolutional Neural Network (CNN),” TensorFlow. [Online]. Available: <https://www.tensorflow.org/tutorials/images/cnn> (accessed Dec. 5, 2025).
- [7] “Image data loading,” Keras. [Online]. Available: https://keras.io/api/data_loading/image/ (accessed Dec. 5, 2025).
- [8] “Adam optimizer,” Keras. [Online]. Available: <https://keras.io/api/optimizers/adam/> (accessed Dec. 5, 2025).
- [9] “Confusion matrix in machine learning,” GeeksforGeeks, Apr. 17, 2024. [Online]. Available: <https://www.geeksforgeeks.org/machine-learning/confusion-matrix-machine-learning/>
- [10] R. Holbrook, “Dropout and batch normalization,” Kaggle. [Online]. Available: <https://www.kaggle.com/code/ryanholbrook/dropout-and-batch-normalization> (accessed Dec. 5, 2025).
- [11] “How to setup TensorFlow/GPU on Windows?,” Reddit, 2024. [Online]. Available: https://www.reddit.com/r/deeplearning/comments/1dq2vol/how_to_setup_tensorflowgpu_on_windows/
- [12] OpenAI, “ChatGPT (May 24 Version),” Large Language Model, 2024. [Online]. Available: <https://chat.openai.com>. Accessed: Dec. 4, 2024. *(Used for brainstorming ideas and guiding learning; the project does not contain any purely AI-generated material.)*