Deep Learning

880663-M-6

Assignment

Using Deep Learning to Perform Multi-Class Classification on the Lung and Colon Cancer Histopathological Image Dataset (LC25000)

Report by:

Jakob Tjurlik (2121838)

March 2024

1. Problem Definition

The following report describes the application of several convolutional deep learning models on a multi-classification task of image data. The dataset contains 25 000 augmented images of different types of cancer, consisting of 750 original lung tissue images and 500 colon tissue images (Borkowski et al., 2019). The task of the project at hand is to find an optimal model that identifies each image with one of five classes (diagnoses).

2. Exploratory Data Analysis

As part of the pre-processing, the images in the original dataset were resized and converted to NumPy, resulting in a collection of 3D arrays of 120x120 pixels and 3 color channels. A set of 15 randomly selected samples is used to visualize the data in Fig. 1a.

The corresponding labels were of one of the following target values: Colon Benign Tissue, Colon adenocarcinoma, Lung Benign Tissue, Lung Squamous Cell Carcinoma, and Lung adenocarcinoma. The values were converted into arrays of binary values using one-hot encoding. Fig. 1b demonstrates the balanced distribution of classes in the dataset.

The data was then split into three stratified sets of training, validation, and test sets of sizes 15 000, 5000, and 5000, respectively.

3. Results of the Baseline Model

The baseline model implements a standard CNN setup of two convolutional layers (with padding) and two MaxPooling layers, followed by two dense layers and an output layer. The latter uses *softmax* as an activation function. The metrics used to evaluate models throughout the whole project are accuracy and categorical cross-entropy loss.

The baseline model ran a test accuracy of 0.768 and a test loss of 0.6171. Fig. 2.1 reveals that the model reaches best accuracies on the last epoch of the model, indicating the ongoing convergence and possibly the need for more training. There is also a noticeable disparity between the train and the validation set results, which suggests that the model is overfitting. Fig. 2.2 shows multi-class ROC curves for validation and test sets.

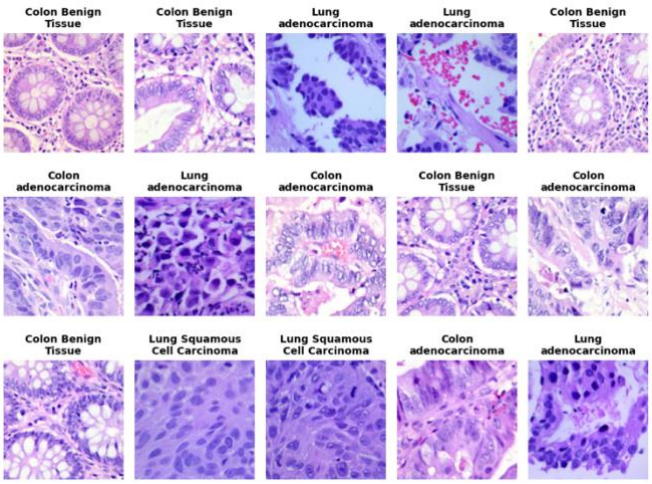

Figure 1: (a) Visualization of random samples from the dataset, with corresponding labels.

Figure 1: (b) Target class distribution, with sample counts.

From the named figures it is evident that there is a discrepancy in how accurately the model predicts different classes. It is also seen in Fig. 2.3 depicting confusion matrices for true and predicted labels. While class 2 is predicted with an F1 score of 0.96 on the test set, other classes are predicted remarkably worse, having F1 scores of 0.60, 0.61, 0.74, and 0.63, respectively. Overall, the model has an average precision of 0.72, an average recall of 0.71, and an F1 score of 0.71. The results on the validation set are almost identical.

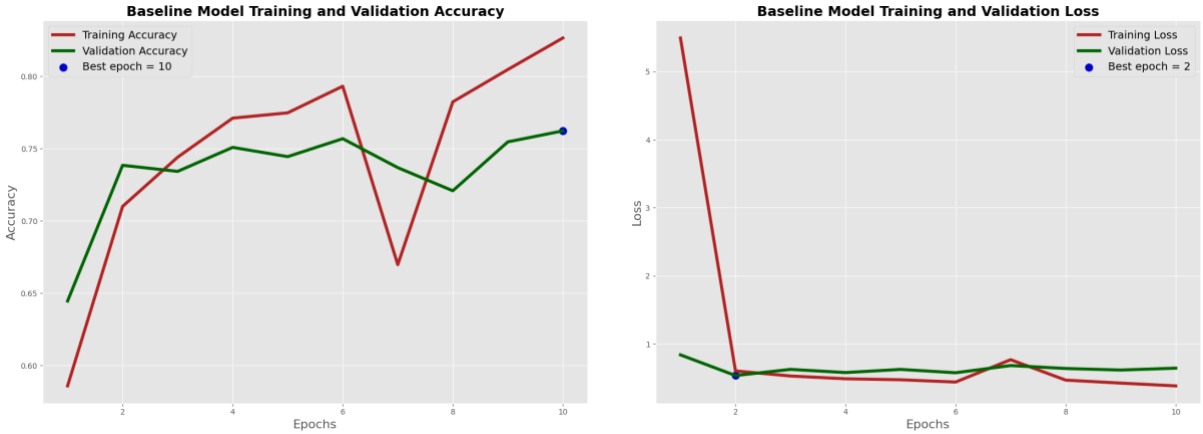

Figure 2.1: (a) Training and validation set accuracy of the baseline model through the epochs; (b) Training and validation set loss of the baseline model through the epochs.

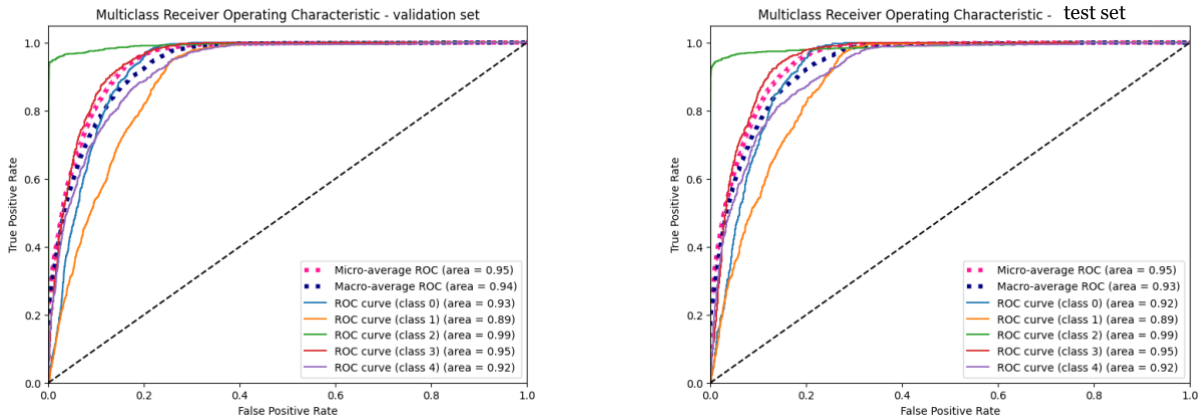

Figure 2.2: (a) ROC curves of the baseline model evaluated (a) on the validation set; (b) on the test set.

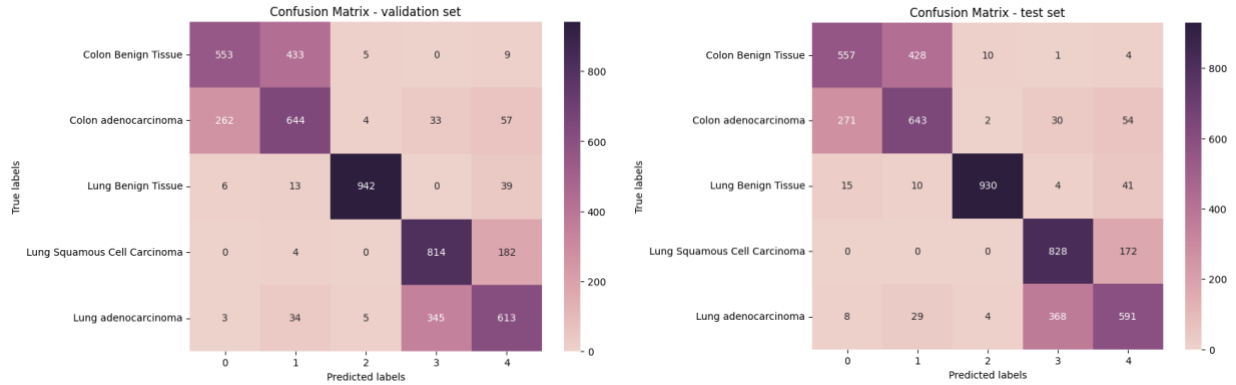

Figure 2.3: (a) Confusion matrices of the baseline model evaluated (a) on the validation set; (b) on the test set.

4. Improved (Fine-tuned) Model and Its Results

The enhanced model consists of four convolutional layers, each followed by a MaxPooling layer of size 2x2. The first convolutional layer has 32 filters of size 5x5, with a stride of 2x2. Each subsequent convolutional layer has a size of 3x3, the default stride of 1, and follows a pattern of increasing number of filters with each layer (64, 128, and 256 respectively). All convolutional layers utilize a ReLU activation function, padding, and L2-regularization with a parameter of 0.00001.

These are followed by two fully connected layers with 128 and 32 neurons respectively, each employing the ReLU activation function and L2-regularization. The output layer employs the *softmax* activation function and similarly includes the L2-regularizer. The model is compiled using the Adam optimizer and the evaluation metrics specified earlier. The model is trained on 20 epochs with a batch size of 128. A summary of the model's structure can be found in the Jupyter Notebook file of the project under the subsection 'Compile the Model'.

The described model achieves a test accuracy of 0.972 and a test loss of 0.1009. This is an improvement of 0.204 over baseline accuracy and of 0.5162 over baseline loss. On the test set, the precision, recall, and F1 scores of different classes are between 0.94 and 1.00. All three metrics have a score of 0.97 on the macro-average and weighted-average levels. Figs. 3.2 and 3.3 demonstrate that the model successfully addresses the

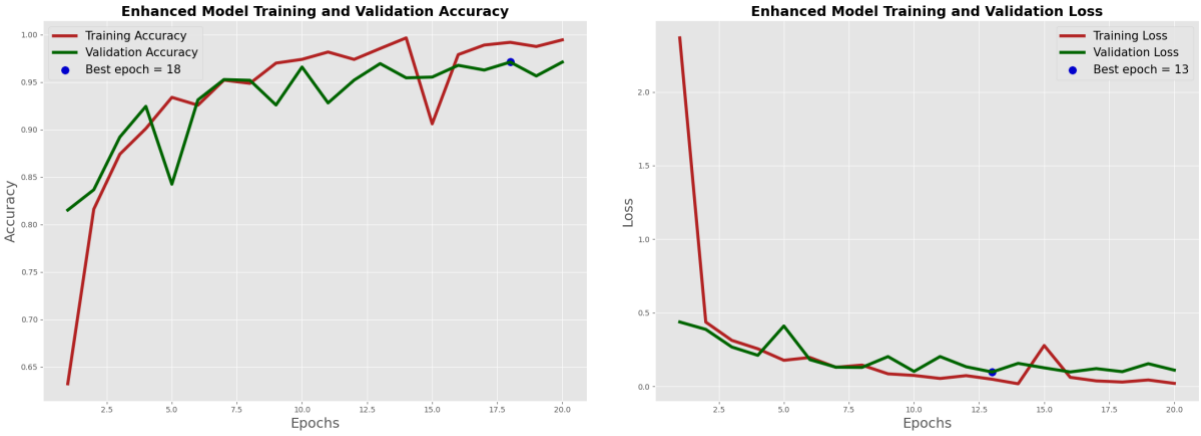

Figure 3.1: (a) Training and validation set accuracy of the improved model through 20 epochs; (b) Training and validation set loss of the improved model through 20 epochs.

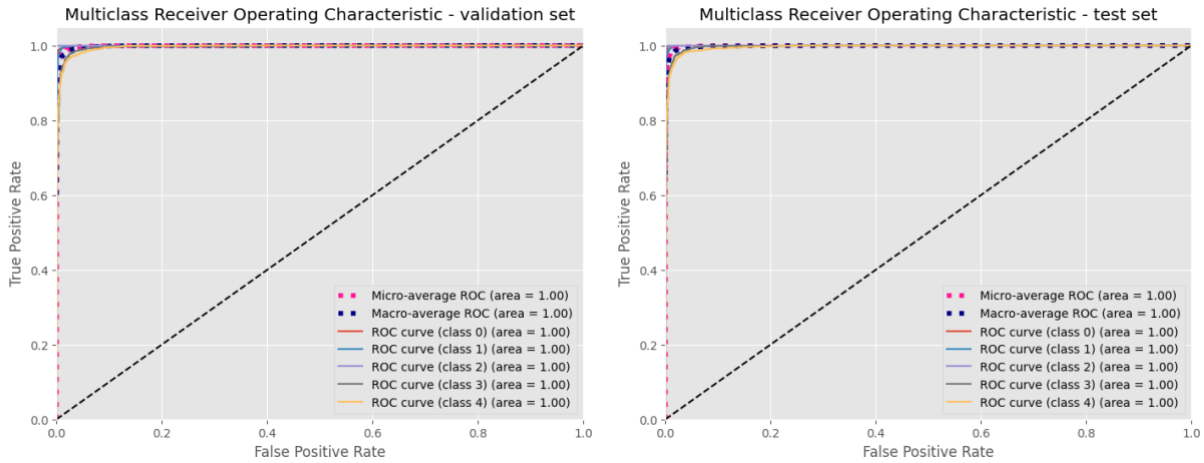

Figure 3.2: (a) ROC curves of the improved model evaluated on the validation set; (b) ROC curves of the improved model evaluated on the test set.

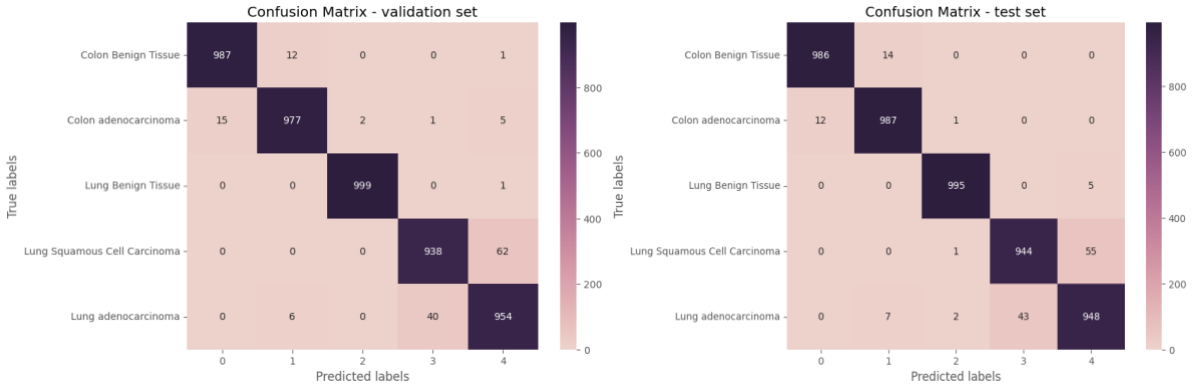

Figure 3.3: Confusion matrices of the improved model evaluated (a) on the validation set; (b) on the test set.

misclassification of classes 0 and 1, as well as 3 and 4 – the issue that was apparent in the baseline model. The model achieves ROC areas of 1.00 on both the validation and the test sets. The model does not overfit, as demonstrated by the Fig. 3.1.

Justifications for the choice of parameters during the tuning process are discussed in more detail in Section 6.

5. Transfer Learning Model and Its Results

The advanced transfer learning model implemented in the task uses the VGG16 architecture. In contrast to a standard VGG16 model, the advanced model is extended by an additional dense layer and a dropout layer. Consecutively, it is comprised of a VGG16 base layer, two dense layers with 256 and 128 neurons respectively, and a dropout layer with a rate of 0.4 between them. Both dense layers utilize the ReLU activation function. The model is compiled with Adam optimizer and is trained for 20 epochs with a batch size of 64. A model summary can be found under the 'Compile the Model' subsection of the Notebook file.

The model similarly demonstrates a noticeable improvement over the baseline model with the test accuracy and loss reaching 0.9782 and 0.0719, respectively. The transfer learning model slightly exceeds the improved model in performance. As shown in Fig. 4, the improvement over the latter is minimal. The VGG16 model seems to converge more quickly and smoothly than the improved model but trains almost ten times longer (approx. 30 s per epoch, compared to the latter's 3 s per epoch).

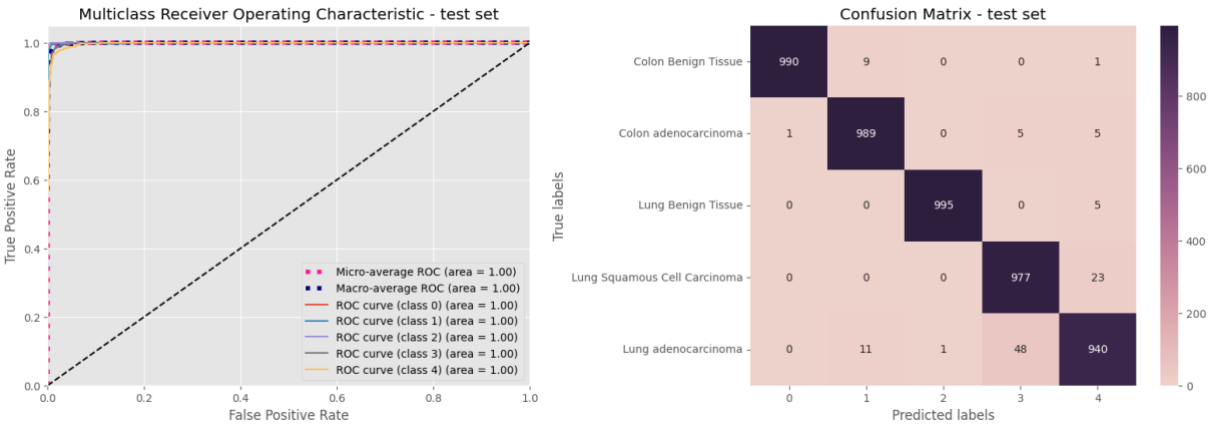

Figure 4: (a) ROC curves of the extended VGG16 model evaluated on the test set; (b) Confusion matrix of the extended VGG16 model evaluated on the test set.

6. Discussion

The results of the improved model show that this architecture can be successfully used in image classification tasks. The baseline model suffers from relatively poor accuracy, particularly evident in pair-wise misclassifications of images of the same organ (for example, classes 0 and 1 depicting colon tissues). The following discussion documents how making reasonable changes to the model addresses these issues, and what might be improved going forward.

The improved model doubles the amount of convolutional and pooling layers of the baseline. The **addition of layers** was the first modification to the model considered. While the baseline model had a test accuracy of 0.768, extending it by one Conv2D and a MaxPooling layer increased the score to 0.8324. Adding another set of convolutional and pooling layers (4 in total) brought the accuracy even higher, to 0.9614. With the addition of hidden layers, we increase the depth of the network, allowing the model to capture more complex patterns – the baseline model is unable to do that.

The extended model also employs a different **number of filters** on each convolutional layer. While the baseline model has a higher number of filters in the first layer (128) and lower after (64), the improved model shows a better performance on a set of filters of the scheme 32-64-128-256. This allows the model to extract simple, low-level features more quickly at first, and then transition to learning more complex details in every next layer. The technique of increasing the number of filters with each layer is also used by Hamed et al. (2023) and Ibrahim (2023). It is, however, important to note that the model's performance improves only slightly with this change.

A more impactful modification on the layer level of the improved model is increasing the **filter size** of the first layer. Instead of a baseline 3x3-sized first kernel, the first layer of the model has filters of a size of 5x5. While this by itself does not improve the model's performance, combining it with a stride of 2x2 does affect the computational efficiency, halving the average training time per epoch. Still, further experimentations with kernel sizes, strides, and padding are to be considered. For instance, Hamed et al. (2023) expand the first kernel to 11x11 with a 4x4-stride and valid padding, which allows it to capture more global features of the image in the initial layer.

For the training process, two alterations are proposed. Firstly, the baseline **batch size** of 32 is increased to 128. It allows the model to train faster and converge more quickly (Devansh, 2022). While a batch size of 64 was also explored (similar to Masud et al., 2023, and Garg and Garg, 2020), the batch size of 128 shows a faster and smoother convergence, without any loss in generalization. A model with **batch normalization** layers is also tried out, as suggested by Masud et al. (2023). Adding batch normalization should stabilize and accelerate the learning process (Goodfellow et al., 2016, p. 316). In the context of this task, due to a significant drop in performance after initial applications, the batch normalization was not tested any further nor used eventually.

Secondly, the **number of epochs** is increased to 20 for the final model and is also utilized for the subsequent transfer learning model. The need for this extension is evident from the baseline results, where the validation accuracy is achieved only on the last epoch. Ibrahim (2023) and Mangal et al. (2020) suggest training on 20 epochs, while Ibrahim and Talaat (2022) use 22. Increasing the number of epochs could be considered in combination with early stopping. Masud et al. (2021) trained the model on 500 epochs, but the test accuracy converged around the 100th epoch, whereas the training accuracy kept on rising. Employing early stopping in models with more epochs could help balance the model performance against the risk of overfitting.

The improved model addresses the issue of overfitting by employing an **L2-regularizer** on its filters. The parameter of the regularizer is changed from the default 0.01 to 1e-5, to achieve a smoother convergence (Brownlee, 2020). The **dropout** methods of rates 0.2 and 0.4 are also tested (as in Ibrahim and Talaat, 2022, and Hamed et al., 2023, respectively). The dropout of 0.4 decreases the training time of the model noticeably, and results in a better performance result for the validation data than the training data. Despite that, the technique is not implemented in the improved model as more pair-wise misclassifications between classes (0 and 1, 3 and 4) emerge. Changing the rate of the dropout does not improve the situation either, hence the L2-regularizer is used. A combination of the two regularization techniques mentioned could be implemented in this task in the future (Goodfellow et al., 2016, pp. 261-262).

Finally, several **optimizers** were tested on the described model. RMSprop (Masud et al., 2021) and Adamax (Ibrahim, 2023) are suggested in the previous literature. Both optimizers also contribute to a high-performance score in the model at hand. Yet the baseline Adam optimizer is selected eventually due to a marginally better score. Adam is also used by Garg and Garg (2020) and Ibrahim and Talaat (2022) and is recommended by Schmidt et al. (2021). Generally, the tuning process shows that the choice of optimizer – one with adaptive learning rates, at least – does not have a significant impact in this context. Goodfellow et al. (2016, p. 306-307) similarly claim there is "no single best algorithm" within this group of optimizers. Additionally, a lower learning rate (schedule) of 1e-5 is tried on Adam, but the convergence is too slow. In further experiments, a more extensive fine-tuning of the learning rates should be considered, as it would help find the most robust convergence method for this task.

Other methods could be relevant in improving the task at hand. In the pre-processing phase, image normalization would potentially help with the model's convergence (e.g., Wadekar and Singh, 2023). More data augmentation would also result in an even larger dataset and potentially better generalization (Goodfellow et al., 2016, p. 236). Ibrahim and Talaat (2022) propose a more complex system of image enhancement, enabling higher quality of the input data. In the training phase, the model could benefit from other activation functions in the hidden layers. The project at hand did not tweak this parameter as the previous literature utilizes the ReLU activation function almost unanimously (e.g. Mangal et al., 2020; Masud et al., 2021; Wadekar and Singh, 2023).

The results of this project show that the model's predictive qualities improve as the model gets more complex. Out of the three presented models, the VGG16 model seems to produce the most accurate and stable results. Ibrahim (2023) reaches a test accuracy of 98.6% with a model resembling VGG16 in structure (Zhang et al., 2015), with multiple convolutional layers stacked together. However, as seen earlier, VGG models tend to require noticeably more time for training than other available deep learning methods (supported by Hamed et al., 2023). The improved model presented in this report reaches similar results and can detect cancerous tissues accurately, whilst requiring less computational power than the VGG16 model. While it can effectively tell apart a malignant lung tissue from a benign one, there is room for improvement when it comes to classifying images of the colon tissues.

Going forward, more hybrid models could be tested on this task. Hamed et al. (2023) propose a structure that uses CNN for feature extraction, followed by a Light Gradient Boosting Model (LightGBM) for classification. Their model reaches an accuracy of 99.6% and is computationally very effective. However, their test results are based on the classification of the lung tissues only. Ibrahim and Talaat (2022) employ an enhanced version of the EfficientNet structure with 7 blocks of layers. They reach an accuracy of 99.5%, testing on all five classes.

7. References

Borkowski, A., Bui, M., Thomas, L., Wilson, C., DeLand, L., Mastorides, S. (2019) 'LC25000 Lung and colon histopathological image dataset'. Available at: https://academictorrents.com/details/7a638ed187a6180fd6e464b3666a6ea0499af4af (Accessed: 27 February 2024).

Brownlee, J. (2020) 'How to Use Weight Decay to Reduce Overfitting of Neural Network in Keras', *Machine Learning Mastery*, 25 August. Available at: https://machinelearningmastery.com/how-to-reduce-overfitting-in-deep-learning-with-weight-regularization/ (Accessed: 9 March 2024).

Devansh (2022) 'How does Batch Size impact your model learning', *Geek Culture*, 17 January. Available at: https://medium.com/geekculture/how-does-batch-size-impact-your-model-learning-2dd34d9fb1fa (Accessed: 9 March 2024).

Garg, Satvik and Garg, Somya (2020) 'Prediction of lung and colon cancer through analysis of histopathological images by utilizing Pre-trained CNN models with visualization of class activation and saliency maps', *AICCC 2020: 2020 3rd Artificial Intelligence and Cloud Computing Conference*, Kyoto Japan: ACM, pp. 38–45.

Goodfellow, I., Bengio, Y. and Courville, A. (2016) *Deep Learning*. MIT Press.

Hamed, E.A.-R. *et al.* (2023) 'An Efficient Combination of Convolutional Neural Network and LightGBM Algorithm for Lung Cancer Histopathology Classification', *Diagnostics*, 13(15), p. 2469.

Ibrahim, A.W. (2023) 'Lung-Colon Cancer | CNN |98.6%', *Kaggle*. Available at: https://www.kaggle.com/code/abdallahwagih/lung-colon-cancer-cnn-98-6 (Accessed: 8 March 2024).

Ibrahim, N.Y. and Talaat, A.S. (2022) 'An Enhancement Technique to Diagnose Colon and Lung Cancer by using Double CLAHE and Deep Learning', *International Journal of Advanced Computer Science and Applications*, 13(8).

Mangal, S., Chaurasia, A. and Khajanchi, A. (2020) 'Convolution Neural Networks for diagnosing colon and lung cancer histopathological images', *ArXiv*, abs/2009.03878.

Masud, M. *et al.* (2021) 'A Machine Learning Approach to Diagnosing Lung and Colon Cancer Using a Deep Learning-Based Classification Framework', *Sensors*, 21(3), p. 748.

Schmidt, R.M., Schneider, F. and Hennig, P. (2021) 'Descending through a Crowded Valley - Benchmarking Deep Learning Optimizers', in M. Meila and T. Zhang (eds) *Proceedings of the 38th International Conference on Machine Learning*. PMLR (Proceedings of Machine Learning Research), pp. 9367–9376.

Wadekar, S. and Singh, D.K. (2023) 'A modified convolutional neural network framework for categorizing lung cell histopathological image based on residual network', *Healthcare Analytics*, 4, p. 100224.

Zhang, X. *et al.* (2015) 'Accelerating Very Deep Convolutional Networks for Classification and Detection', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10), pp. 1943-1955.