EAI-6010
Applications of Artificial Intelligence

BY INSTRUCTOR
Abhijit Sanyal

PROJECT REPORT
Module 2

SUBMITTED BY
Iti Rohilla

On
Nov-14-2024

Computer Vision Classification Report

Pneumonia Detection Using a Customized CNN in PyTorch

1. Introduction

This report presents the process and findings from a computer vision classification task aimed at detecting pneumonia using chest X-ray images. The objective was to train a convolutional neural network (CNN) on the Pneumonia Detection dataset, which consists of X-ray images labeled as either "Normal" or "Pneumonia." The CNN was implemented using PyTorch and trained with various techniques to optimize accuracy while avoiding overtraining. Additionally, the performance of the model was analyzed to identify the best- and worst-performing classes.

2. Dataset Selection and Parameter Justification

The Pneumonia Detection dataset was selected because of its medical relevance in helping detect pneumonia—a serious respiratory condition that can lead to severe health complications if untreated. Early and accurate detection through chest X-rays can significantly aid healthcare providers in diagnosis and intervention. The dataset, consisting of labeled X-ray images as either "Normal" or "Pneumonia," is structured for binary classification, which aligns well with the goal of detecting specific patterns indicative of pneumonia versus normal conditions.

3. Model Selection and Customization

For this task, I chose to build a custom CNN architecture in PyTorch instead of using a pre-trained model. The CNN architecture consists of two convolutional layers, each followed by ReLU activation and max-pooling. A dropout layer was included to reduce overfitting by randomly dropping nodes during training. The final layers are two fully connected layers, with the second output layer configured for binary classification (Normal or Pneumonia). The model was trained using the Adam optimizer with a learning rate of 0.001 and cross-entropy loss as the loss function.

4. Scenario of Overtraining

Overtraining, or overfitting, occurs when a model performs well on the training data but fails to generalize to unseen data, resulting in poor validation performance. In this scenario, overtraining was observed during the training of the CNN model.

The early epochs showed that the training loss decreased consistently, indicating that the model was learning the features of the training dataset. However, from around epoch 4 onward, the validation loss began to fluctuate, with no consistent decrease. Specifically, after epoch 6, the training loss continued to improve, but the validation loss did not follow this trend, indicating the model was fitting too closely to the training data while failing to generalize to new images. This is a classic sign of overtraining, as the model captures noise or redundant patterns in the training dataset rather than the fundamental patterns that generalize to new data.

5. Training Methods Used to Prevent Overtraining

To prevent overtraining, I implemented several techniques, including dropout layers, data augmentation, and early stopping.

Dropout Layers

Dropout layers were added to the model after each fully connected layer to help reduce overfitting. During training, dropout layers randomly set a fraction of the neurons to zero, which forces the model to learn more robust features rather than relying on specific neurons. I set the dropout probability to 0.3, which significantly helped control the model's tendency to overfit.

Data Augmentation

Data augmentation techniques were applied to the training dataset to increase the variety of images, which helps the model generalize better. The transformations included random horizontal flips and slight rotations, as well as resizing and normalizing images to fit the input requirements of the CNN. These augmentations effectively simulated a larger and more diverse dataset, which made the model more resilient to variations in image orientation and positioning.

Early Stopping

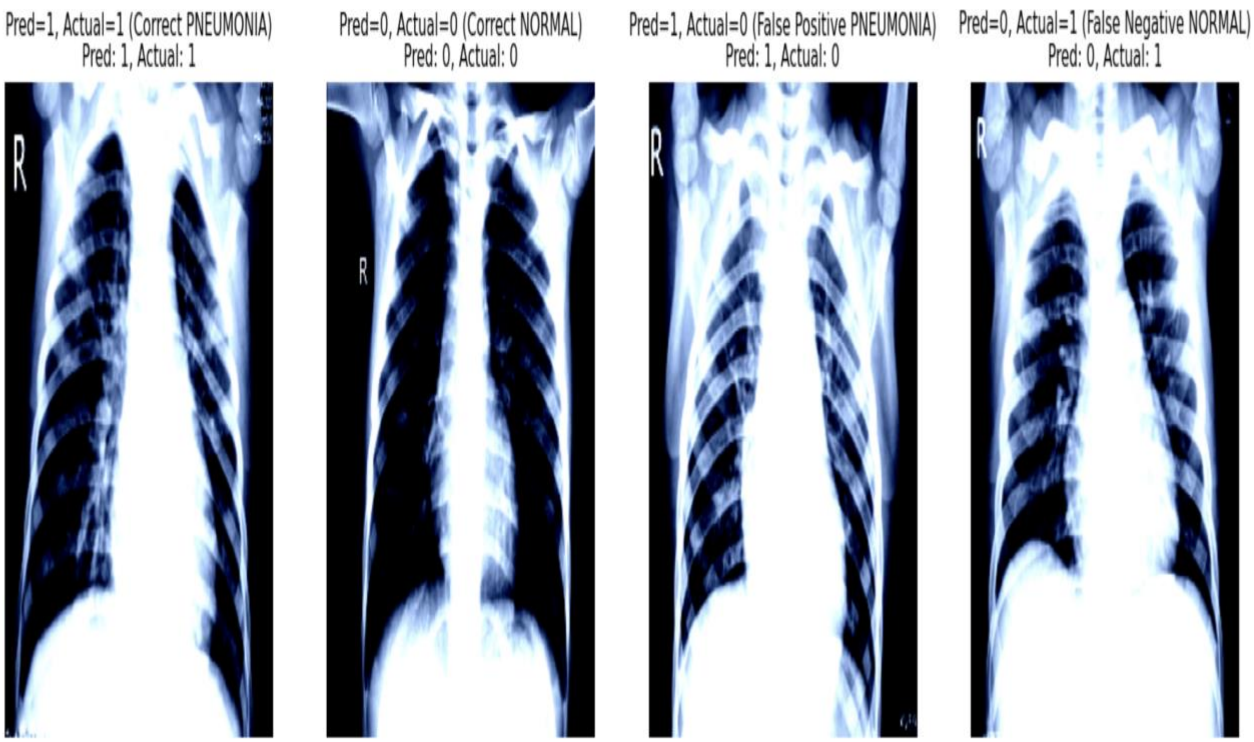
Early stopping was implemented by monitoring the validation loss at each epoch. If the validation loss did not improve for a specified patience period (three consecutive epochs in this case), the training was halted. Early stopping prevented the model from training excessively and minimized the risk of overfitting. In this scenario, early stopping was triggered at epoch 9, suggesting that the model had reached an optimal point and further training would have led to overfitting.

6. Determining When to Stop Training

The implementation of early stopping provided a clear indication of when to stop training. Throughout the training process, the model's validation loss fluctuated and failed to decrease consistently after a certain point. This was especially evident after epoch 6, where the validation loss began to increase slightly despite continued improvement in training loss. By epoch 9, it was clear that the model had reached a point of diminishing returns, and early stopping was triggered to prevent overfitting.

7. Performance Analysis: Best and Worst-Performing Classes

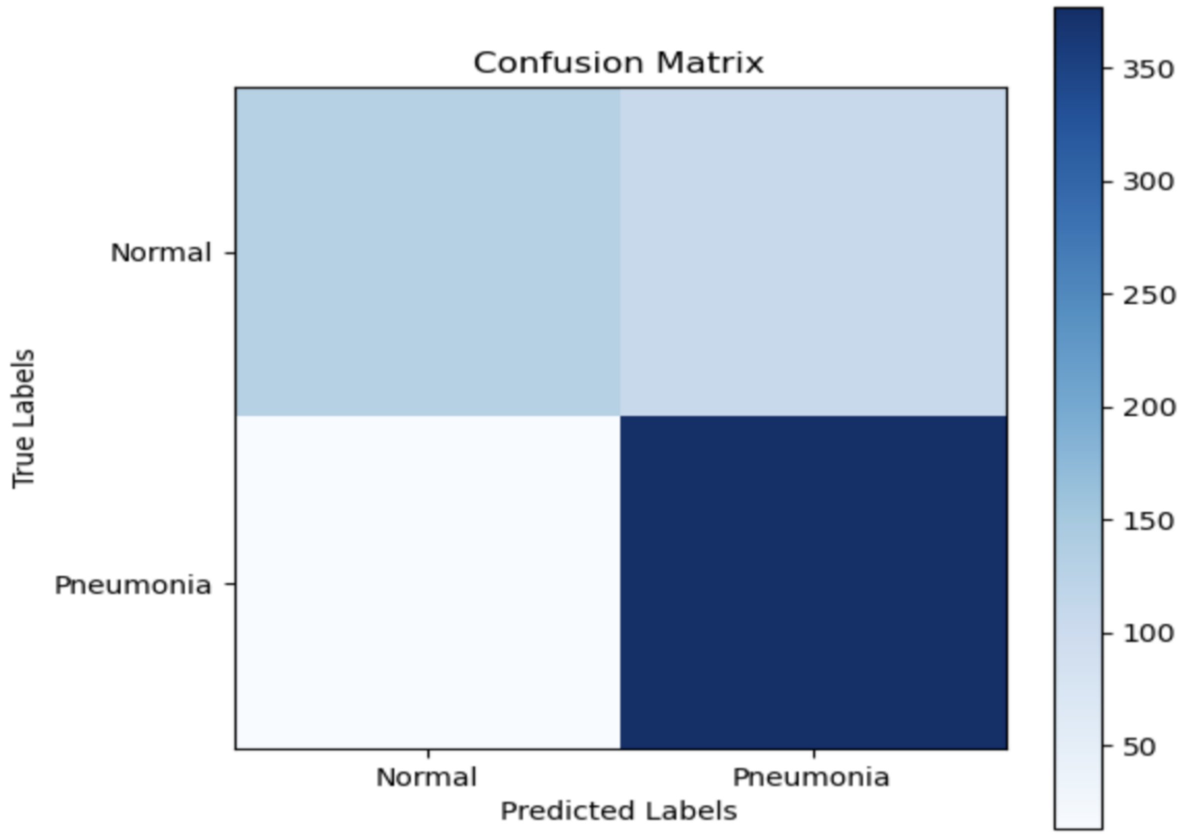
The model's performance was evaluated on a separate test dataset, and the results were analyzed using metrics such as precision, recall, and F1-score. Additionally, a confusion matrix was generated to provide insights into classification accuracy for each class.



Classification Report Summary

Classification Report:

	precision	recall	f1-score	support
Normal	0.91	0.56	0.69	234
Pneumonia	0.78	0.97	0.87	390
accuracy			0.81	624
macro avg	0.85	0.76	0.78	624
weighted avg	0.83	0.81	0.80	624



The model achieved an overall accuracy of 81%, with a high recall for the Pneumonia class (0.97) but a comparatively low recall for the Normal class (0.56). This suggests that the model was more effective at identifying pneumonia cases but struggled to accurately detect normal cases.

Confusion Matrix

The confusion matrix highlights the model’s tendency to classify Normal images as Pneumonia (false positives), indicating a bias toward predicting Pneumonia. This bias may be due to the prevalence of Pneumonia images in the dataset, which could cause the model to favor this class.

8. Recommended Path for Improvement

The model's weaker performance in identifying Normal cases indicates room for improvement. To address this, the following strategies are recommended:

1. **Class Balancing**: The Pneumonia class has a higher representation in the dataset compared to the Normal class, which may contribute to the model’s bias. Balancing the dataset by either oversampling Normal images or undersampling Pneumonia images could help reduce this bias.
2. **Hard Negative Mining**: Implementing hard negative mining could improve performance on Normal images. This technique focuses on samples that the model has previously misclassified, allowing it to learn more effectively from challenging cases.
3. **Using a Pre-Trained Model**: Fine-tuning a pre-trained model like ResNet or VGG on this dataset could improve performance. These models have been trained on large datasets and may extract more relevant features, leading to better generalization.
4. **Hyperparameter Tuning**: Experimenting with different dropout rates, learning rates, and optimizers could improve model performance. Additionally, increasing the patience parameter in early stopping could allow the model to reach a slightly better generalization point.

9. Conclusion

In this project, a custom CNN was designed and trained on the Pneumonia Detection dataset to classify X-ray images as Normal or Pneumonia. Although the model demonstrated good overall performance, overtraining was observed after several epochs, and early stopping was employed to mitigate this issue. Data augmentation, dropout layers, and early stopping proved to be effective in controlling overfitting. The model showed a bias towards the Pneumonia class, which led to lower recall for Normal cases. Strategies like class balancing, hard negative mining, and fine-tuning a pre-trained model are recommended to address these limitations and enhance model accuracy in future iterations.

CITATIONS

- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

  He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). https://doi.org/10.1109/CVPR.2016.90

- Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4700–4708). https://doi.org/10.1109/CVPR.2017.243

- Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning* (pp. 448–456). http://proceedings.mlr.press/v37/ioffe15.pdf

- Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1412.6980

- Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data, 6*(60). https://doi.org/10.1186/s40537-019-0197-0

- Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., & Liang, J. (2018). UNet++: A nested U-Net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support* (pp. 3–11). Springer, Cham. https://doi.org/10.1007/978-3-030-00889-5_1