GPU Optimization with TensorFlow

Student Id: 22093598 **Module: Applied Data Science 2**

Train the network and plot the resulting loss functions and metrics. Create plots that demonstrate the predictive results of the network. Comment on these results in your report.

The initial training of the CNN model on the Oxford-IIIT Pet dataset shows that the model learns gradually, with the training accuracy improving steadily over the epochs. However, the validation accuracy fluctuates and does not consistently improve, indicating potential overfitting, especially as the validation loss increases in later epochs, as shown in plot 1 in the notebook. The early convergence of the training loss compared to the validation loss suggests that the model may memorize the training data rather than generalize well to new data. These results highlight the need for further optimization, such as implementing regularization techniques, tuning hyperparameters, or modifying the network architecture to enhance generalization and prevent overfitting.

Address the performance issues in the image processing pipeline, and comment on your choices in your report.

The initial image processing pipeline had several performance issues, including slow training speed, suboptimal predictive performance, and potential overfitting due to limited data augmentation. To address these issues, we implemented a series of enhancements. Firstly, we included data augmentation techniques such as random rotations, flips, and zooms, which increased the diversity of the training data and helped reduce overfitting. Secondly, we resized images to a consistent size and normalized pixel values, improving the consistency and quality of input data. Lastly, we optimized dataset batching to make more efficient use of computational resources, thereby speeding up the training process. These changes collectively enhanced the training speed and improved the model's ability to generalize to new data, resulting in better overall performance.

Adjust the design of the CNN to achieve improved predictive results, and comment on the choices you make in your report. (10 marks)

Several key adjustments were made to enhance predictive performance and training efficiency in the optimized CNN model. Batch normalization layers were added after each convolutional layer for stabilization and faster training, while dropout layers were included to prevent overfitting. The model's

depth was increased with additional convolutional and pooling layers, improving feature extraction. Convolutional layer kernel sizes were set to 3x3 for a balance of efficiency and capability. A global average pooling layer was introduced to reduce parameters and computational load, followed by a dense layer with batch normalization and dropout for refined feature representation. These changes aimed to create a robust, efficient model with higher accuracy and better generalization.

Change the settings used in training the network to improve both the training speed and predictive results. (5 marks)

We implemented several changes to improve the CNN model's training speed and predictive performance. Firstly, we adopted an exponentially decaying learning rate schedule with the Adam optimizer, starting with an initial learning rate 1e-3. These changes help fine-tune the model by gradually reducing the learning rate. This change ensures the model converges more effectively and prevents overshooting the optimal weights. We also incorporated EarlyStopping and ReduceLROnPlateau callbacks. ReduceLROnPlateau dynamically reduces the learning rate when the validation loss plateaus, allowing the model to fine-tune further. Additionally, we carefully selected a moderate batch size to balance memory efficiency and the ability to capture diverse data patterns. These combined adjustments enhance the model's efficiency and effectiveness, improving predictive performance and better generalization on validation data.

Comment in your report how the changes you made have improved the speed and performance of the network.

The changes to the CNN model architecture and training settings have significantly enhanced training speed and predictive performance. Using an exponentially decaying learning rate with the Adam optimizer ensured dynamic adjustment, starting high for rapid learning and gradually decreasing for fine-tuning, leading to better convergence. EarlyStopping prevented overfitting by stopping training when validation loss stopped improving, saving resources and time. ReduceLROnPlateau further fine-tuned the model by reducing the learning rate when validation loss plateaued. Compared to the initial model, the optimized model shows more stable and rapid convergence in loss and more consistent improvement in accuracy, indicating improved generalization to unseen data. These adjustments resulted in faster training, higher performance, and a more reliable model for practical applications.