Assignment 02- AML
Group 24

Introduction

In this assignment, the objective is to apply convolutional neural networks (convnets) to image data, specifically utilizing the Dogs vs Cats dataset from Kaggle. The primary focus lies in understanding the relationship between sample sizes and the approach of training convnets: either from scratch or by leveraging a pre-trained network.

To explore this relationship, various models were constructed with different training set sizes: 1000, 1500, and 2000 samples, while keeping the test and validation sets constant at 500 samples each. Each model was trained using common parameters: accuracy as the metric, Adam as the optimizer, and binary_crossentropy as the loss function.

Furthermore, to delve deeper into the performance insights, a pre-trained model was employed specifically for the training set with a size of 2000 samples. This pre-trained model utilized the RMSprop optimizer, providing additional perspectives on model performance.

This approach allows for a comprehensive examination of how sample sizes impact the effectiveness of convnets, both when training from scratch and when leveraging pre-existing knowledge through pre-trained networks.

01.Consider the Cats & Dogs example. Start initially with a training sample of 1000, a validation sample of 500, and a test sample of 500 (like in the text). Use any technique to reduce overfitting and improve performance in developing a network that you train from scratch. What performance did you achieve?

Training Sample Size: 1000

Based on the provided code and its results, here's the performance achieved for training a network from scratch on the Cats & Dogs dataset with an initial training sample size of 1000:

- Training Accuracy: Approximately 80.4%
- Validation Accuracy: Approximately 68.1%
- Test Accuracy: Approximately 69.9%

Summary of performance:

- The model exhibits a relatively high training accuracy of approximately 80.4%, indicating that it fits the training data well.
- However, there is a noticeable performance gap between the training and validation accuracies, suggesting some degree of overfitting. The validation accuracy is around 68.1%, which is lower than the training accuracy.

- The test accuracy, which is approximately 69.9%, provides an estimate of how well the model generalizes to unseen data.

02. Increase your training sample size. You may pick any amount. Keep the validation and test samples the same as above. Optimize your network (again training from scratch). What performance did you achieve?

Training Sample Size: 1500

- Training Accuracy: Approximately 89.9%
- Validation Accuracy: Approximately 76.1%
- Test Accuracy: Approximately 74.9%

Summary of performance:

- The model exhibits a relatively high training accuracy of approximately 89.9%, indicating that it fits the larger training data well.
- However, there is still a noticeable performance gap between the training and validation accuracies, suggesting some degree of overfitting. The validation accuracy is around 76.1%, which is lower than the training accuracy.
- The test accuracy, which is approximately 74.9%, provides an estimate of how well the model generalizes to unseen data.

03. Now change your training sample so that you achieve better performance than those from Steps 1 and 2. This sample size may be larger, or smaller than those in the previous steps. The objective is to find the ideal training sample size to get best prediction results.

Training Sample Size: 2000

- Training Loss: Decreased over epochs from approximately 0.2470 to 0.0803.
- Validation Loss: Decreased over epochs from approximately 0.2294 to 0.1537.
- Test Loss: Approximately 0.1704.

Summary of performance:

- The model achieved a lower test loss compared to the previous steps, indicating improved generalization performance.
- The decreasing trends in both training and validation losses suggest that the model effectively learned from the larger training dataset.
- The achieved test loss of approximately 0.1704 indicates good performance in making predictions on unseen data.

- When compared with the previous samples, training sample 2000 is the ideal training sample size.

04. Repeat Steps 1-3, but now using a pretrained network. The sample sizes you use in Steps 2 and 3 for the pretrained network may be the same or different from those using the network where you trained from scratch. Again, use any and all optimization techniques to get best performance.

Training from Scratch with Data Augmentation:

- Trained a CNN from scratch using data augmentation on training sets of 1000, 1500, and 2000 samples.
- The best accuracy achieved on the test set was approximately 67.4%.

Transfer Learning with Pretrained VGG16 Model:

- Used the VGG16 model pre-trained on ImageNet for feature extraction without data augmentation.
- A classifier was added on top of the pre-trained base, and the model was fine-tuned with the RMSprop optimizer.
- The test accuracy achieved was around 97.1%.

Transfer Learning with Pretrained VGG16 Model and Data Augmentation:

- Similar to the previous approach, you used the VGG16 model pre-trained on ImageNet for feature extraction.
- Incorporated data augmentation during training.
- A classifier was added on top of the pre-trained base, and the model was fine-tuned with the RMSprop optimizer.
- The test accuracy achieved was around 97.4%.

Summary

The model is run using various training sizes, both with and without data augmentation, and with a pretrained model (both with and without data augmentation) in the tabular format that is shown below.

Optimizer: Adam					
S.no	Training Set	Validation and Test Size	Data augmentation	Pretrained Model	Loss and Accuracy on Test
1	1000	500, 500	No	No	loss: 0.6272 - accuracy: 0.6990
2	1500	500, 500	No	No	loss: 0.5159 - accuracy: 0.7490
3	2000	500, 500	No	No	loss: 0.5053 - accuracy: 0.7740
4	1000	500, 500	Yes	No	loss: 0.6171 - accuracy: 0.6740
5	1500	500, 500	Yes	No	loss: 0.5832 - accuracy: 0.7280

6	2000	500, 500	Yes	No	loss: 0.8400 - accuracy: 0.9740
7	2000	500, 500	No	Yes	loss: 7.7509 - accuracy: 0.9670
8	2000	500, 500	Yes	Yes	loss: 2.2153 - accuracy: 0.9760
Optimizer: rmseprop					
9	2000	500, 500	No	Yes	loss:7.2070 - accuracy: 0.9710

Overall, the experiment involved training a neural network from scratch on the Cats & Dogs dataset, gradually increasing the training sample size, and then repeating the process using a pretrained network (VGG16) with transfer learning. Here are the key observations and recommendations based on the results.

Training from Scratch

- Performance improved as the training sample size increased.
- However, there was noticeable overfitting, with the validation accuracy lagging behind the training accuracy.
- Increasing the sample size improved performance, but the gap between training and validation accuracies persisted.

Transfer Learning with Pretrained VGG16

- Significantly outperformed training from scratch, achieving higher test accuracies.
- Utilizing a pretrained model and fine-tuning the classifier led to better generalization.
- Data augmentation further improved performance, highlighting the importance of leveraging additional techniques for better results.

Recommendations

- **Transfer Learning**: Given the superior performance of transfer learning over training from scratch, it's recommended to leverage pretrained models like VGG16 whenever possible, especially when working with limited data.
- **Data Augmentation**: Incorporating data augmentation during training can effectively mitigate overfitting and improve model generalization.
- **Sample Size**: While increasing the training sample size improved performance to some extent, the gains were not as significant as those achieved through transfer learning. Therefore, focusing on techniques like transfer learning and data augmentation may yield better results than solely increasing the dataset size.
- **Validation Strategy**: Monitoring the gap between training and validation accuracies is crucial for detecting overfitting. Techniques such as early stopping can help prevent overfitting by halting training when validation performance begins to degrade.
- **Optimization Techniques**: Experimenting with different optimization techniques, learning rates, and model architectures can further enhance performance.