My Interpretations and Recommendations:

I would like to draw attention to several aspects that have enhanced the model's performance and others that have the potential to further improve it

1. Fine-tuning proved to be highly effective in achieving optimal accuracy when combined with dropout and data augmentation. This technique involves utilizing pre-learned model representations to extract features for a new challenge, resulting in improved performance.

2. When looking for publications about various forms of data augmentation, I tried out the intriguing technique of **color augmentation**. With the help of this, we can enhance brightness, contrast, or saturation. I think some legacy images can be categorized using this.

3. During the training process, it took 61 seconds to complete one Epoch when the training sample was around 6000. However, when the training sample size was increased to 10,000, it took 64 seconds to complete one Epoch.

   If there is proper infra (GPU), then I think this can be handled.

4. Using optimisers effectively and increasing the sample size has improved accuracy.

Other observation: Using optimizers effectively and increasing the sample size has improved accuracy. Optimizers are algorithms that help in minimizing the error function during training. The choice of optimizer can have a significant impact on the model's performance. Therefore, selecting an appropriate optimizer and setting its parameters correctly is crucial. Moreover, the report suggests that increasing the sample size also helps in improving accuracy. This is because larger datasets provide more diverse and representative samples, enabling the model to learn better and generalize well to new data.

Additionally, used **NVIDIA Deep Learning GPU Training System (DIGITS) with** the current dataset for classifying images. The tool is reported to have an effective model designed with the best optimization techniques embedded in it. suggests that the processing time for classification was very less and the accuracy claimed by the tool was 98%. I suggest that other peers should try this tool for effective understanding.

It implies that optimizing the model through proper selection of optimizers, increasing the sample size, and using effective tools like DIGITS can contribute to improved accuracy in image classification tasks. These findings can be helpful for researchers and practitioners in the field of deep learning and computer vision.

Q1) Training sample of 1000, a validation sample of 500, and a test sample of 500.

Test accuracy: 71.6 Loss: 0.5

Unregularized Method: The test accuracy in the unregularized model is around 71.6 with ADAM optimizer. After a certain period, the model appears to be overfitting and may not have generalized to the new data.

Here, I've utilized Just Data Augmentation without a dropout value or regularization method. defining a new network that just uses data augmentation. I was curious to know how accuracy would be defined by merely employing data augmentation. In the following steps, I'll experiment with combining augmentation with various regularization strategies.

Q2) Increasing the training size and keeping test, validation size the same:

Here, I have tried to induce the following things.

- Dropout
- Learning rate
- Early Stopping Monitor

When optimization was no longer helpful, I employed early halting to end it. I determined a value of 10 for patience.

Test Accuracy: 81%, which was superior to Augmentation alone. Hence, I have noticed that by using these two and expanding the sample size, I was able to improve my results while marginally lowering the loss. The regularized model appears to have a little bit higher accuracy compared to the unregularized model.

Q3) Changing the training sample so that we achieve better performance than those from Steps.

I have increased the training sample here, keeping the same validation and test sample sizes.

Test Accuracy: 90.7 loss: 0.33

This produced a better result for me than the earlier model.

Q4) Repeat Steps 1-3, but now using a pretrained network.

Test Accuracy: 96.7%

VGG16 Pretrained Convnet Network: Here are a few conclusions I was able to draw while comparing a model created from scratch versus one utilizing a pre-trained network.

To Conclude

The test's findings are clear: employing a pre-trained model for various picture recognition tasks is advantageous for several reasons.

1. Using a pre-trained model requires less training and requires less effort in building the model's architecture.

2. Using a pre-trained model can be a more accurate approach than building a custom-built convolutionary neural network (CNN) from scratch. Pre-trained models are deep neural networks that have already been trained on a large dataset, usually with millions of images, and have learned to identify common features and patterns in the data. These pre-trained models have already gone through the computationally expensive process of training and have learned a set of complex image features that are useful for many image recognition tasks.

Using a pre-trained model for a specific image recognition task is known as transfer learning. With transfer learning, a pre-trained model can be fine-tuned on a smaller dataset specific to the task, allowing it to learn additional features that are relevant to the new dataset. This fine-tuning can significantly improve the accuracy of the model, as it has already been learned to recognize many of the features present in the new data set.

On the other hand, building a custom-built CNN from scratch requires a large dataset and significant computational resources to train the model. Additionally, the model may not perform as well as a pre-trained model due to the limited amount of data available for training, and the lack of pre-learned features in the model.

Therefore, using a pre-trained model and fine-tuning it for a specific task can often lead to higher accuracy and faster development times than building a custom-built CNN from scratch.

3. The size of the dataset used for training plays a significant role in the accuracy of the model. When we train a model from scratch, using a large sample size improves the learning process. On the other hand, when we use a pre-trained model, we can improve the accuracy by unfreezing a few of the top layers of a frozen model and then training the newly added part of the model along with the pre-trained layers. This process is called fine-tuning. By fine-tuning a pre-trained model, we can take advantage of the previous learning and adapt it to the new data, which helps to achieve better results compared to training a custom-built model from scratch. The pre-trained models have already learned the general features from large amounts of data, making it easier and faster to learn new features from a smaller dataset. Fine-tuning allows us to leverage the generalization power of pre-trained models to improve the accuracy of our specific task.