National Tsing Hua University
Fall 2023 11210IPT 553000
Deep Learning in Biomedical Optical Imaging
Homework 4

WISARUD YONGBANJERD[1]
Student ID:109003888

1. Task A: Model Selection

For the transfer learning tasks, I would select the following two pre-trained models from ‘torchvision.models’:

1.1 ResNet-50

This model strikes a good balance between accuracy and complexity. It's based on the concept of residual learning to ease the training of deep networks. Its performance has been extensively validated on ImageNet.

Architecture: ResNet-50 is part of the Residual Networks family, which was a breakthrough in deep learning. It has 50 layers deep and uses skip connections or shortcuts to jump over some layers. These shortcuts or residual connections allow the model to avoid the problem of vanishing gradients, as they provide alternative pathways for the gradient during backpropagation.

Pre-trained Performance: On ImageNet, which has 1000 classes, ResNet-50 achieves a top-1 accuracy of around 76.15% and a top-5 accuracy of 92.87%. These results demonstrate the model’s high capability of feature extraction for a variety of classes.

Computation Time: Although not the lightest model available, ResNet-50 is considerably more efficient than its deeper counterparts like ResNet-101 or ResNet-152. It can be a good trade-off between performance and computational resources for transfer learning tasks.

1.2 MobileNetV2

It's designed to be lightweight and efficient, making it a good candidate for applications where computational resources are limited, like mobile or embedded devices.

Architecture: MobileNetV2 is built on an inverted residual structure where the input and output of the residual block are thin bottleneck layers opposite to traditional residuals. The intermediate expansion layer uses lightweight depth wise convolutions to filter features as a source of non-linearity.

Pre-trained Performance: This model is slightly less accurate on high-level benchmarks like ImageNet compared to larger models such as ResNet-50, but it is designed for mobile and resource-constrained environments. It still achieves a respectable top-1 accuracy of about 71.8% and a top-5 accuracy of 91%.

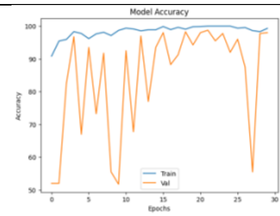
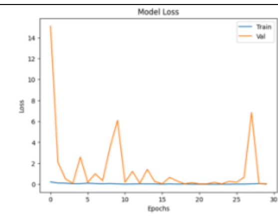
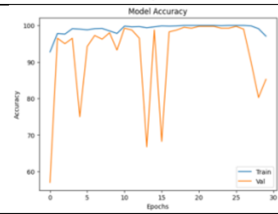
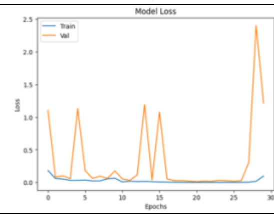
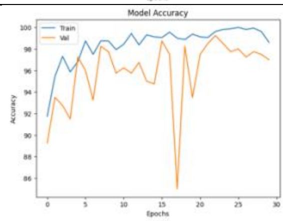
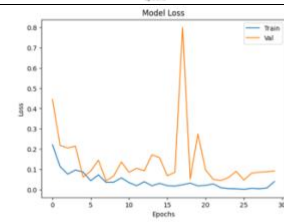
Computation Time: MobileNetV2 is designed to be fast and efficient. It has a smaller number of parameters compared to other models like ResNet-50, which makes it faster in terms of inference time. This is crucial for biomedical optical imaging tasks where the models might need to be deployed on handheld or portable devices with limited computational capacity.

2. Task B: Fine-Tuning the ConvNet

2.1. ResNet-50

I am introducing two adjustments namely drop out technique and adjust learning rate in order to increase the performance of the model. Here is the performance from the adjustments as shown on Table 1.

Table 1. Showing results from four different techniques with Resnet-50 Model

			Test Accuracy
Running The Restnet-50			73.5%
Adjust Learning Rate from 1e-3 to 5e-4			81.25%
Apply Drop Out (0.5)			80.75%

From the condition 1, the discrepancies between training and validation accuracy and loss suggest that the model could be overfitting to the training data. The model performs well on the data it has seen but is not generalizing well to the validation set. The test accuracy of 73.5% provides a baseline performance for the model. While it is decent, the volatility in the validation metrics suggests that the model's performance might not be consistent across different sets of unseen data.

Compared to the first condition, the second setup with a reduced learning rate shows signs of less overfitting. The validation metrics are following the training metrics more closely. Lowering the learning rate seems to have stabilized the learning process, resulting in fewer spikes in validation loss and more consistent accuracy. An increase in test accuracy to 81.25% is a significant improvement over the 73.5% from the initial condition. This suggests that the model with the adjusted learning rate is better at generalizing to unseen data.

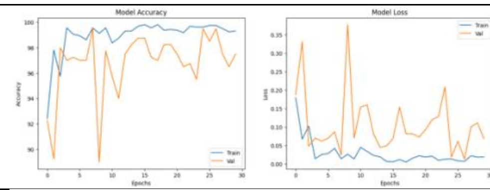
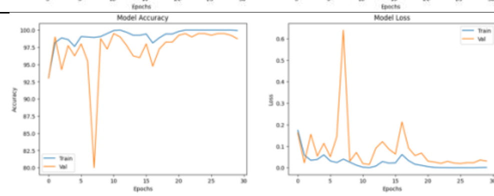
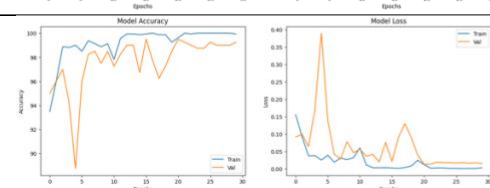
Incorporating dropout seems to have had a regularizing effect, as the gap between training and validation accuracy is less pronounced than in the first condition. Dropout forces the model to learn more robust features by randomly dropping units, which should help in generalizing better to unseen data. There is less evidence of overfitting with dropout compared to the initial setup. The model does not reach the extreme highs of training accuracy seen previously, which is expected as dropout prevents co-adaptation of features. The test accuracy is 80.75%, which is slightly lower than the model with the adjusted learning rate (81.25%) but higher than the original model (73.5%). This slight decrease from the adjusted learning rate condition could be due to the regularizing effect of dropout leading to a slightly more conservative model.

Overall, fine-tuning the model with both an adjusted learning rate and dropout improved performance, with learning rate adjustment seeming to have the most substantial impact. However, the slight decrease in test accuracy from Condition 2 to Condition 3 might suggest that while dropout has a regularizing effect, it may also slightly hinder the model's ability to achieve the highest accuracy possible in this specific scenario.

2.2. MobileNetV2

I am introducing two adjustments namely drop out technique and learning rate scheduler in order to increase the performance of the model. Here is the performance from the adjustments as shown on Table 2.

Table 2. Showing results from different techniques with MobileNetV2 Model

		Test Accuracy
Running The MobileNetV2		82%
Apply Drop Out (0.2)		73.5%
Implement Learning Rate Scheduler (ReduceLROnPlateau)		79.5%

From Table 2., condition 1 has high discrepancy between training and validation accuracy, as well as the volatile validation loss, indicate that the model has not generalized well. This could be due to overfitting.

For the condition 2, training accuracy remains high but shows some fluctuations, indicating a bit of instability in the learning process. It stays above 97.5% for most epochs. Validation accuracy also shows fluctuations and a gap from training accuracy, although it appears slightly more stable compared to the first condition. Overall, the training loss decreases and remains low, while validation loss is higher and more volatile, similar to the first condition. While dropout is meant to improve generalization, the significant difference between training and test accuracy shows that the model is still learning patterns specific to the training data that do not transfer well to the test data. The instability in the learning process and the lower test accuracy, when compared to the first condition, suggest that this particular setup is less effective at adapting to the new task and requires further optimization.

In the third condition, the training accuracy quickly reaches a very high level and fluctuates slightly but remains around or at 100% after about 10 epochs. The validation accuracy also increases rapidly but it shows more variability and generally trends below the training accuracy, albeit by a small margin in later epochs. There's a gap between training and validation accuracy which suggests some degree of overfitting. The training loss decreases steadily and approaches zero, indicating the model is fitting the training data very well. The validation loss decreases but has spikes that correspond to the fluctuations in the validation accuracy. The fact that the validation loss does not consistently rise suggests that while the model may be overfitting slightly, it still generalizes well.

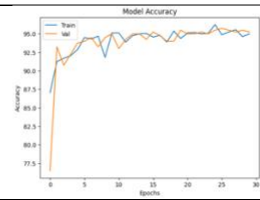
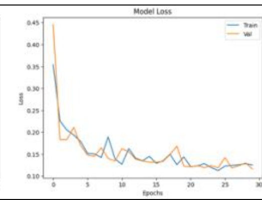
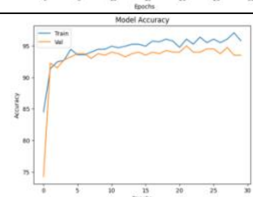
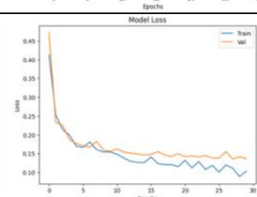
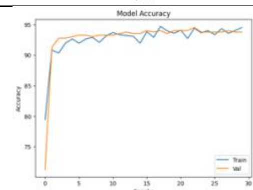
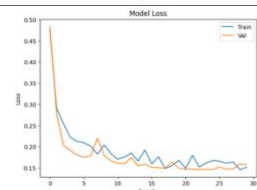
In summary, the model with the learning rate scheduler seems to have learned the training data to near perfection, but the gap in performance on the test set indicates a mismatch between what the model has learned and what is necessary to perform well on unseen data.

3. Task C: ConvNet as Fixed Feature Extractor

3.1. ResNet-50

I am introducing two adjustments namely drop out technique and adjust learning rate in order to increase the performance of the model. Here is the performance from the adjustments as shown on Table 3

Table 3. Showing results from different techniques with Resnet-50 Model with fixed feature Extractor.

			Test Accuracy
Running The Resnet-50			85.75%
Adjust Learning Rate from 1e-3 to 5e-4			85.75%
Apply Drop Out (0.5)			85%

The first condition has effectively learned from the training data, reflected by high training accuracy. The volatility in validation accuracy and the gap between training and validation accuracy may indicate overfitting. The model shows a good ability to generalize, but the noted fluctuations suggest that it may benefit from additional strategies to improve stability, such as learning rate annealing or more advanced regularization methods.

Unlike the initial fluctuations seen in the first condition configuration's early epochs, this configuration demonstrates a steady increase in accuracy from the outset. The loss curves for the second condition are smoother compared to the first condition, which could suggest that the model with the adjusted learning rate is learning more steadily. Both conditions seem to be equally effective in terms of final accuracy, but the path to convergence is different. The first condition starts with higher accuracy, indicating a faster adaptation, while the second condition shows a more gradual learning process. The second condition exhibits a steady learning curve, indicating that the lower learning rate may allow for a more measured adaptation to the new task, despite the initial lower accuracy.

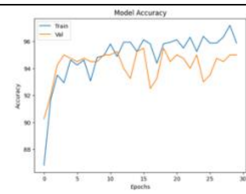
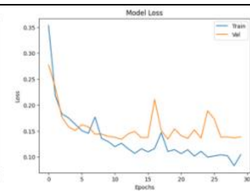
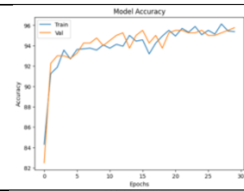
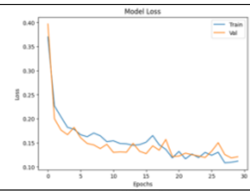
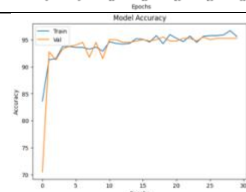
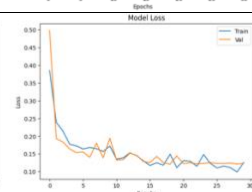
Introducing dropout has led to a stable convergence, with less fluctuation in validation accuracy compared to the previous models. This can help in improving the model's generalization ability by reducing overfitting. There is a slight drop in test accuracy (85%) compared to the previous configurations (85.75%). This small difference could be due to the dropout's regularization effect, which sometimes can slightly decrease performance on the training set while aiming to improve generalization on unseen data.

All three conditions seem to generalize well, but the third condition might offer the most robustness against overfitting due to the inclusion of dropout, which is not fully reflected in the test accuracy but could be beneficial in real-world applications.

3.2. MobileNetV2

I am introducing two adjustments namely drop out technique and learning rate scheduler in order to increase the performance of the model. Here is the performance from the adjustments as shown on Table 4.

Table 4. Showing results from different techniques with MobileNetV2 Model with fixed feature Extractor

		Test Accuracy
Running the MobileNetV2	 	78.5%
Apply Drop out (0.2)	 	83.25%
Implement Learning Rate Schedule (ReduceLROnPlateau)	 	82.25%

From Table 4., the first condition, the fixed feature extractor model has a good training accuracy, but the validation metrics indicate room for improvement in model generalization. Given the model reaches a test accuracy of 78.5%, the high variability in the validation loss might not be as critical as it could suggest that although the model has some epochs where it performs poorly on the validation set, overall, it generalizes reasonably well.

From the second condition, the introduction of dropout appears to have had a positive impact on the model's performance, evidenced by the tighter accuracy and loss curves. Achieving a test accuracy of 83.25% is an improvement over previous iterations (78.5%), suggesting that dropout has helped the model generalize better to unseen data.

From the third run shows a high degree of accuracy and generalization, with a tight correlation between training and validation metrics. This is indicative of a well-performing model. The test accuracy reported is 82.25%, which is slightly lower than the second run but still within a high-performance bracket.

It appears that using fixed feature extractors from the MobileNetV2 model can indeed produce strong results on a new task. The original pre-trained features are quite powerful on their own, as seen in the first condition. However, incorporating dropout (second condition) has proven to be the most effective single technique for improving generalization in this particular set of experiments, as it yields the highest test accuracy. The learning rate scheduler also contributes positively to performance (third condition) but doesn't reach the same level as dropout when considered alone.

4. Task D: Comparison and Analysis

4.1. ResNet-50

The models with fixed feature extractors (FFE) demonstrated better generalization to the validation set, which can be particularly advantageous when dealing with small datasets or when the training and validation distributions are highly similar. Fixed feature extractors offered more stability in the learning process, with less fluctuation in the validation metrics. This stability makes training more predictable and can simplify the process of model selection. Fine-tuning all layers in the model without FFE showed a higher risk of overfitting, which necessitates careful tuning of regularization parameters and learning rates. In contrast, FFE models, by design, are more robust against overfitting due to their limited capacity for change.

4.2. MobileNetV2

The MobileNetV2 models without fixed feature extractors have demonstrated superior performance in terms of test accuracy. This indicates that allowing the model to adjust its learned features to the specifics of a new task can significantly enhance its adaptability and effectiveness. The need for techniques like dropout and learning rate schedulers is apparent in both scenarios but serves different purposes. For fixed feature extractors, these techniques mainly help to prevent overfitting of the classifier on top of the pre-trained features. For models without fixed feature extractors, they aid in fine-tuning the process to converge to a better set of weights without overfitting. The gap in performance between fine-tuned models and those with fixed feature extractors may vary depending on the nature of the task and the similarity between the new task and the data the model was originally trained on. If the new task is closely related to the original training data, fixed feature extractors might perform comparably well. But if the tasks are quite different, fine-tuning can leverage the model's capacity to adapt its features more extensively.

The choice between fine-tuning and using a fixed feature extractor should be guided by the nature of the dataset and the task. Fine-tuning offers higher adaptability at the cost of a greater risk of overfitting and requires careful tuning. FFE provides better generalization and stability, which can be beneficial when the new task does not deviate significantly from the pre-trained tasks or when data is limited.

In practice, the best approach often lies in a combination of the two: starting with a fixed feature extractor to benefit from the established features and then fine-tuning a select number of layers once the model has been sufficiently adapted to the new data.

5. Task E: Test Dataset Analysis

From my point of view, transfer learning is like taking shortcuts from a map you already know to help you navigate a new one. But if the new map is too different from what you've learned before, those shortcuts might not work as well, and you could hit a dead end in improving your performance. Sometimes, the model gets too good at noticing things in the training data and misses the mark when it sees new data it hasn't learned from before. To prevent this, you can mix up the training with different techniques or double-check the model's work with new examples. However, it's tricky to get this mix just right.

On top of that, choosing the best settings for your model is really tough—it's like trying to find one specific piece in a huge pile of Legos. You need to pick the right pieces that fit your model just right, which can take a lot of trial and error. All these challenges are part of the process of training your model to understand and do well on new tasks.

References

1. Pytorch Contributors. (2023). MODELS AND PRE-TRAINED WEIGHTS. https://pytorch.org/vision/main/models.html#table-of-all-available-quantized-classification-weights
2. Ng, A. (n.d.). Deep Learning Specialization. Coursera. Retrieved [November 4, 2023], from https://www.coursera.org/specializations/deep-learning
3. Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? In Advances in Neural Information Processing Systems (pp. 3320-3328). Neural Information Processing Systems Foundation.