Topic: Develop Second Modeling Approach Alice Tang and Yinda Chen

TF Lite Breast Cancer Detection Week 7: Second Model Approach

1. Model Approach

Last week, we tried building our own Convolutional Neural Network (CNN) for our model, but while it ran without issues, the validation loss and accuracy were abysmal. This week, we switched to transfer learning, using a pre-trained ResNet50 model to detect breast cancer in mammography images. ResNet50, a deep CNN pre-trained on the ImageNet dataset, captures a wide variety of features. By freezing some of its layers and adding our custom fully connected layers, we fine-tuned the model for classifying mammograms as benign or malignant. We also believe the model performed significantly better than last week's due to the integration of data augmentation techniques like random flipping, brightness adjustment, and contrast and saturation manipulation— which improved generalization and reduced overfitting—especially with our imbalanced dataset. Additionally, we drew inspiration from Kaggle user "hayder17"'s notebook for preprocessing, which had a notable positive impact on our model's performance, which further highlights how proper preprocessing is crucial.

1.1 Rationale for the Model Choice

Our decision to use ResNet50 can be attributed to its proven capability in image classification tasks, and its ability to detect subtle patterns in medical images. Our dataset has a decent amount of images, but it still lacks in comparison to the large number of parameters in deep neural networks. Utilizing transfer learning with a pre-trained model effectively enables us to use pre-existing knowledge of image features, thus improving performance and speeding up convergence. By freezing a large portion of the pre-trained model, we are also able to retain the important low-level features that were learned from the larger ImageNet database— simultaneously fine-tuning the upper layers for our specific task of mammogram classification. The customizability and flexibility of implementing this model is a compelling reason to choose this pre-trained model. Three versions of the ResNet50 model were tested in evaluating tradeoffs between complexity and performance, where adjustments were made to the number of unfreezed layers, dropout rates, and dense layer sizes for improved optimization.

1.2 Complexity of the Modeling Approach

The complexity of the model can be controlled by balancing the number of trainable layers in the ResNet50 base model and the structure of the custom layers added. In our initial model (try_model), 50% of the ResNet50 layers were unfrozen, thereby allowing more complex feature learning, while allowing more complex feature learning, while a dense layer of 1024 units and a dropout rate of 0.5 were added to the custom top layers. In our second version (try_model_v2), 60% of the ResNet50 layers were unfrozen, with a reduced dense layer size of 512 units and a lower dropout rate of 0.3 as a test for reducing model complexity. In the third model (try_model_v3), the complexity was increased by unfreezing 70% of the ResNet50 layers, allowing for deeper feature extraction from the pre-trained model. To balance this complexity, we reduced the dense layer size to 256 units, and increased dropout rate to 0.7. The higher dropout rate was implemented to cultivate better generalization, due to the larger number of trainable layers. Each version of the model used categorical cross-entropy as the loss function, Adam optimizer for efficient convergence, and accuracy, precision, and recall metrics for evaluation.

Additionally, considering a proper augmentation strategy and data balancing method was important in properly handling the dataset's class imbalance, where the proportion of malignant cases is much smaller. To mitigate any problems, we augmented images in the smaller class, ensuring the model does not become biased towards benign cases. This week's data augmentation increases the overall complexity of the model compared to last week, but this is crucial in improving our model.

Through leveraging a pre-trained architecture like ResNet50, along with special modifications for this dataset, we strive to strike a careful balance between model complexity and performance.

2. Evaluating the Hyperparameters

While developing the three distinct ResNet50 models, there were several hyperparameters we evaluated to optimize model performance and ensure effectiveness in training. The goal of this evaluation was to determine the best hyperparameter setting for the dataset when comparing performance metrics like accuracy, precision, and recall.

Model 1: ResNet50 with 50% of Layers Unfrozen

For our first model, 50% of the ResNet50 layers were unfrozen, meaning half the model will be fine-tuned during training. We set the following hyperparameters:

- **Base Model:** ResNet50 (pre-trained on ImageNet)
- **Unfrozen Layers:** 50%
- **Dense Layer:** 1024 neurons with ReLU activation
- **Dropout Rate:** 0.5
- **Optimizer:** Adam with a learning rate of 1e-4
- **Loss Function:** Categorical cross-entropy
- **Metrics:** Accuracy, Precision, Recall

We trained the model for 7 epochs, and made sure the results of this model could be compared with the others based on the validation accuracy, precision, and recall.

Model 2: ResNet50 with 60% of Layers Unfrozen

For our second model, we made the following adjustments:

- **Unfrozen Layers:** 60%
- **Dense Layer:** Reduced to 512 neurons with ReLU activation
- **Dropout Rate:** Reduced to 0.3
- **Optimizer:** Adam with a lower learning rate of 5e-5
- **Metrics:** Accuracy, Precision, and Recall (separately tracked as precision_v2 and recall_v2)

We also trained this model for 7 epochs, and logged the performance metrics for comparison.

Model 3: ResNet50 with 70% of Layers Unfrozen

In our third (and last) model, we made the following changes:

- **Unfrozen Layers:** 70%
- **Dense Layer:** Reverted back to 1024 neurons
- **Dropout Rate:** Set to 0.4
- **Optimizer:** RMSprop with a learning rate of 1e-4
- **Metrics:** Accuracy, Precision, and Recall (tracked as precision_v3 and recall_v3)

Once again, this model was trained for 7 epochs and we included validation on the same dataset.

2.1 Hyperparameters Evaluated

To evaluate the performance, we must compare the accuracy, precision, and recall for all three models of training. A few key points of comparison include:

- **Impact of Freezing/Unfreezing Layers:** We gradually unfroze more layers from 50%, 60%, to 70%, which permits more feature representations to be learned in each later model. From this, we can garner whether fine-tuning more layers improves performance.
- **Impact of Dense Layer Size:** The Dense layer size was reduced in the second model and reinstated in the third. We will examine if there is an effect on learning based on this.
- **Impact of Dropout Rate:** The first model used a high dropout rate of 0.5, while the second and third models had lower dropout rates. We will see if lowering dropout rates improves or reduces generalization.
- **Impact of Optimizer and Learning Rate:** The third model used a different optimizer (RMSprop) and similar learning rate to the first model, differing with the second model's lower learning rate. From this, we can evaluate how these changes influenced training stability and final performance.

Our expectation is that one of these three configurations will clearly outperform the others in terms of generalization (validation accuracy), and robustness (precision/recall). The results will be examined later in this document.

3. Model Performance Metrics

For evaluating the performance of our models, there were three key metrics we focused on: accuracy, precision, and recall.

3.1 Accuracy

Accuracy measures the proportion of the correct predictions (both true positives and true negatives) of all predictions. It is often used as a metric, but may not be sufficient as the only metric in imbalanced datasets. In our case, accuracy can allow us to understand the overall performance of the models, but may not be as reliable when classes are imbalanced.

3.2 Precision

Precision shows how many of the model's predicted positives were actual positives. It is especially important in cases where minimizing the false positives is crucial, since it shows the accuracy of the model's positive predictions. This is a good metric to choose, since reducing the number of false positives is crucial when having a wrong positive prediction could lead to serious repercussions.

3.3 Recall

Recall (or the sensitivity/true positive rate) indicates how many actual positives the model identified correctly. Similar to precision, recall is a good metric for the other scenario where false negatives could be costly or harmful. Since our problem may involve identifying rare positive cases, failing to predict these cases may even be more detrimental than a false positive. Obtaining a high recall means that the model can catch as many positive cases as possible.

In general, the metrics we chose for this evaluation include accuracy for an overall assessment, but with special consideration to precision and recall due to the importance of understanding the trade-off between false positives and false negatives. Through these metrics, we'll be able to gain a better understanding of how the models perform in real-world scenarios, especially when the cost of making an incorrect classification is uneven. Each of these metrics will be tracked for the 3 different models (v1, v2, v3) and the metrics will allow us to determine the best results for the given task.

4. Analysis of the Training and Validation Metrics Across Variations

As mentioned before, we adjusted the hyperparameters and created 3 variations of the Resnet50 model.

Model 1

Model 1 starts with an initial accuracy of 0.5385 and a loss of 0.9464. By the first epoch, there is significant improvement, reaching an accuracy of 0.7044 and a validation accuracy of 0.8245. The loss steadily decreases, which shows that the model is learning effectively. From the beginning to the end of each epoch, the training and validation metrics show a clear upward trend

- **Accuracy** improves to 0.9790 by epoch 7.
- **Validation Accuracy** stabilizes around 0.9607.
- **Loss** decreases consistently with the final validation loss at 0.0966.
- **Precision** and **Recall** both improve similarly, starting at 0.5385 and reaching 0.9790, aligning well with accuracy, exhibiting fewer false positives and false negatives.

We're off to a good start with this model. This model shows a strong capability to generalize well across training and validation data, though there is a slight divergence in the last epoch that may signify the initial signs of overfitting.

Model 2

Model 2 begins with a slightly better accuracy of 0.6154 compared to Model 1, and the training process shows rapid improvements in accuracy.

- **Accuracy** reaches 0.9831 by epoch 7, and **Validation Accuracy** also improves to 0.9679.
- **Loss** continues to drop, finishing at 0.0450 for training and 0.0918 for validation, indicating minimal overfitting.
- **Precision** and **Recall** are closely realigned, which indicates strong performance across both metrics with a final value of 0.9831— reflecting accurate and sensitive predictions.

Model 2 shows there is efficient learning from the start and only slightly outperforms Model 1 in the final validation metrics.

Model 3

Onto our last model! Model 3 starts with a solid performance at Epoch 1, where the accuracy is 0.6970 and validation accuracy is 0.7332. However, Model 3 exhibits instability across epochs:

- By the last epoch, the model reaches accuracy of 0.9862, but Validation Accuracy fluctuates more than the previous models, finishing at 0.9471.
- The loss begins high at 0.6916 and ends at 0.0534 for training, but validation loss shows inconsistency, with the final validation loss at 0.2212.
- Even though precision and recall improve over time, the random behavior of validation loss may prove there is some possible overfitting, as shown through the lower validation accuracy in the middle epochs (Epoch 3–5).

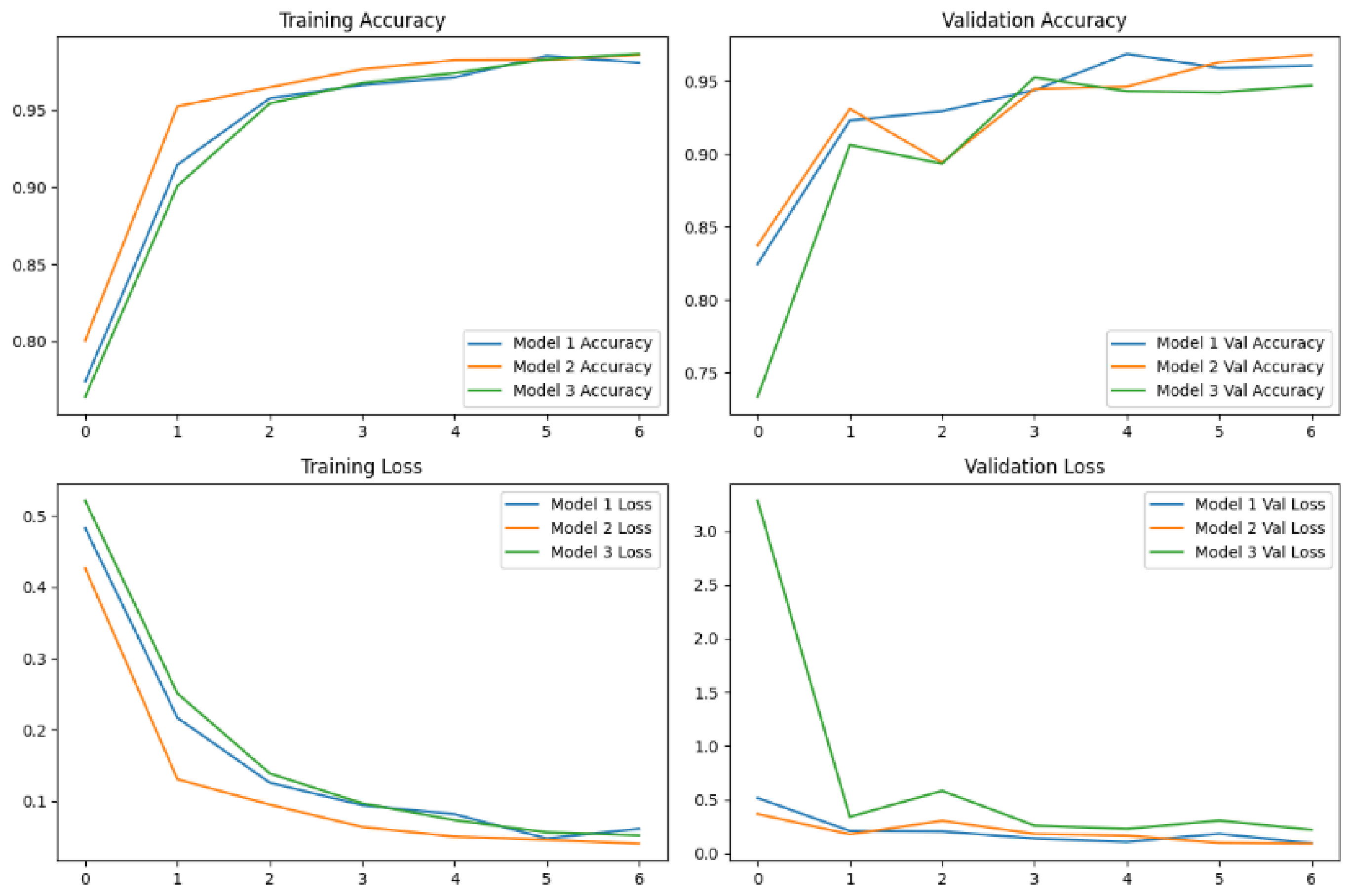
Overall, Model 3 presents with strong overall accuracy and precision, but the instability in validation loss suggests it may not generalize as well as Models 1 and 2, especially on unseen data

4.1 Comparison Table of Each Model

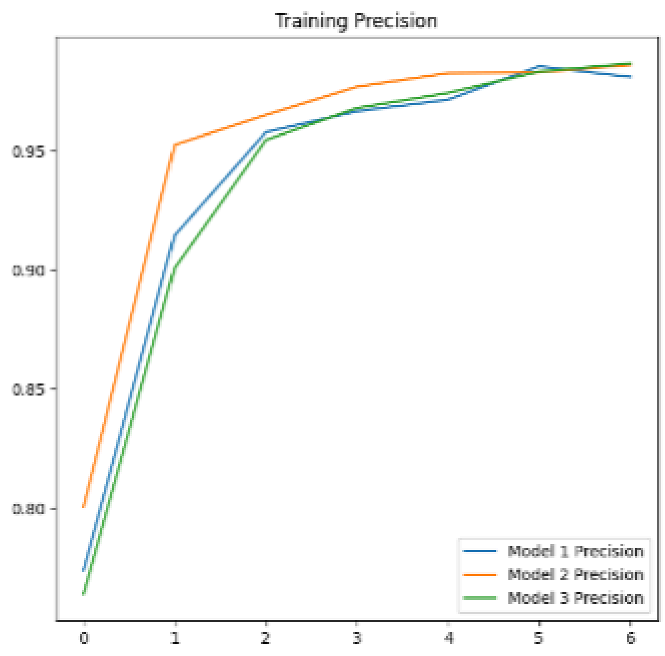
Model	Epoch	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
Model 1	1	0.780858696	0.825320542	0.480371535	0.437261522
Model 1	2	0.906976759	0.940705121	0.22311388	0.163657889
Model 1	3	0.962880135	0.932692289	0.109034218	0.214752361
Model 1	4	0.96481812	0.951121807	0.098495863	0.138146803
Model 1	5	0.976296961	0.922275662	0.069653891	0.210704148
Model 1	6	0.980471075	0.956730783	0.057326391	0.184525281
Model 1	7	0.980918288	0.953525662	0.059124365	0.132300138
Model 2	1	0.796362579	0.911858976	0.422733426	0.226265743
Model 2	2	0.952146709	0.934294879	0.12578465	0.202672794
Model 2	3	0.967799664	0.889423072	0.090422623	0.408461362
Model 2	4	0.975104332	0.96714741	0.066133775	0.102012508
Model 2	5	0.982259989	0.943910241	0.045915235	0.168080285
Model 2	6	0.984496117	0.943108976	0.048927348	0.203147575
Model 2	7	0.987775803	0.943108976	0.033306196	0.172981501
Model 3	1	0.763565898	0.81089741	0.517337084	0.798546314
Model 3	2	0.900566459	0.884615362	0.252175629	0.498230457
Model 3	3	0.951699436	0.874198735	0.142279014	0.760936916
Model 3	4	0.968992233	0.938301265	0.102396503	0.408288568
Model 3	5	0.973315418	0.930288434	0.080401078	0.350371599
Model 3	6	0.980620146	0.948717952	0.061201528	0.269799083

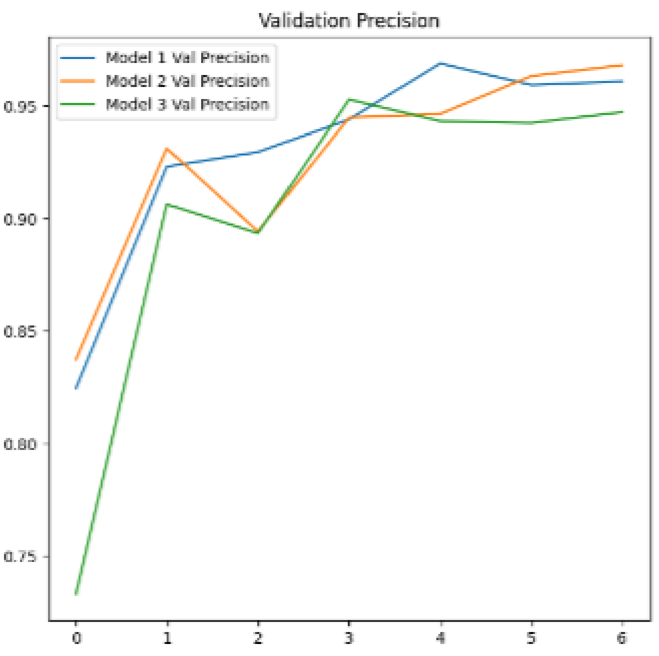
Model 3	7	0.982856274	0.944711566	0.052912932	0.480846703

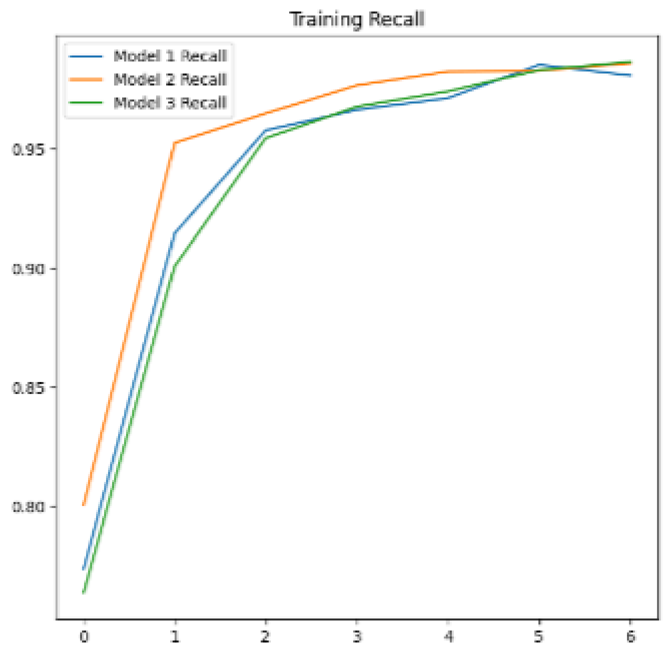
4.2 Training and Validation Accuracy of Each Model

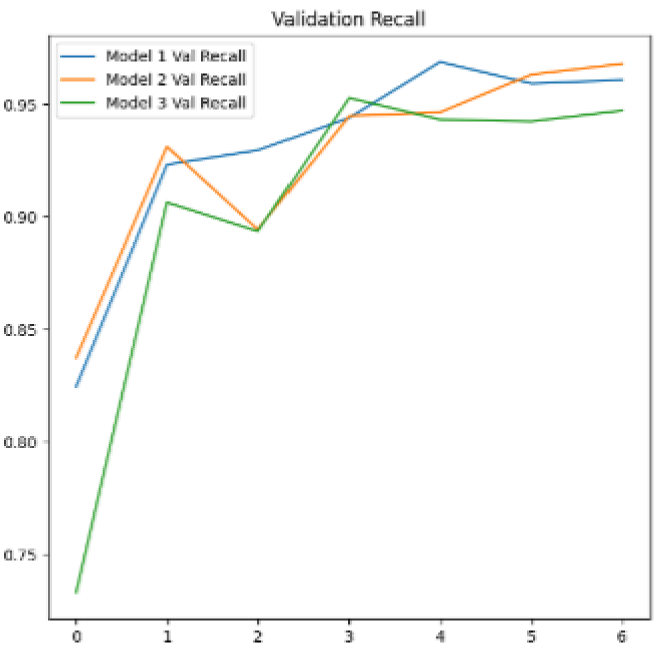


4.3 Training and Validation Precision and Recall of Each Model









5. Model of the Week: Best Performance

Based on the training and validation metrics across all 3 models, Model 2 is our best-performing model for this week. Here is the justification as to why:

- **Strong and Stable Performance:** Model 2 consistently exhibited high accuracy, precision, and recall across all epochs. With an initial accuracy of 73.6% in the first epoch, it quickly improved to over 98% by the final epoch, with similar findings in precision and recall.
- **Validation Metrics:** The validation accuracy of Model 2 reached 96.79%, which was the highest across all models. With a loss value of 0.0918, this indicates better generalization to the validation data in comparison to the oher models with higher validation loss. Model 2 had strong validation precision and recall throughout, which further signifies the robustness of the model.
- **Lower Validation Loss:** Even though Model 1 and Model 3 still had solid performance, Model 2's validation loss values consistently stayed lower than the others— even in later epochs. Overall, this shows better generalization and stability during training, in turn, making it less likely to overfit compared to Model 3, which experienced a high initial validation loss of 3.2818 in the first epoch.

Overall, Model 2 displayed the most optimal balance between accuracy, precision, recall, and validation loss, making it our best model of the week.