graphs

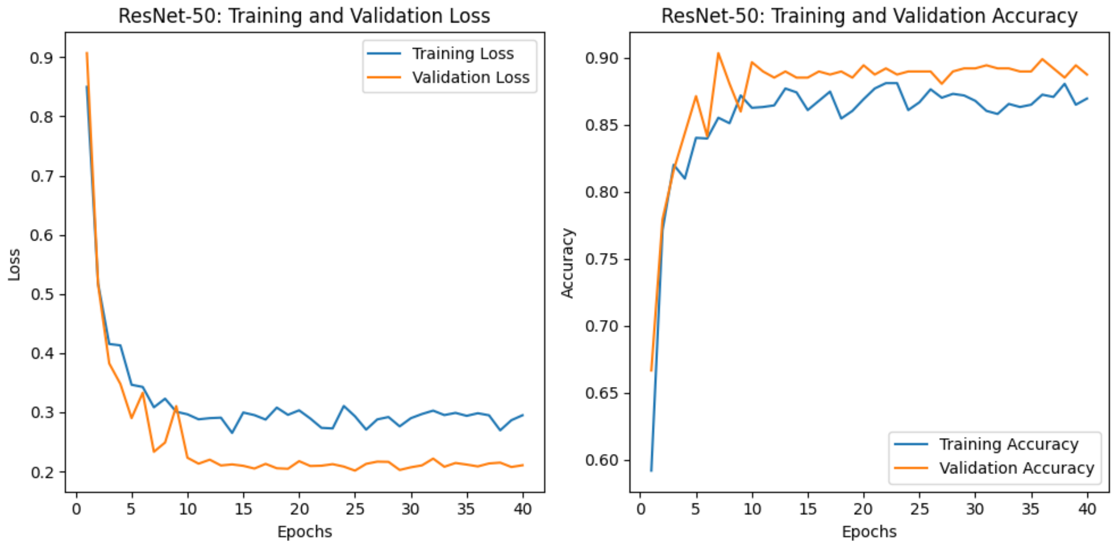
May 7, 2024

[2]:
```
import matplotlib.pyplot as plt

# Define the training and validation loss and accuracy values for AlexNet
alexnet_train_loss = [0.7972, 0.5677, 0.5274, 0.4818, 0.4716, 0.4244, 0.4344, 0.
 ↪4460, 0.4059, 0.4069, 0.4042, 0.4047, 0.4037, 0.3798, 0.4002, 0.3873, 0.
 ↪3679, 0.3500, 0.3649, 0.3376, 0.3566, 0.3544, 0.3390, 0.3436, 0.3637, 0.
 ↪3232, 0.3280, 0.3407, 0.3454, 0.3300, 0.3004, 0.3133, 0.3029, 0.3128, 0.
 ↪3340, 0.3187, 0.3023, 0.3045, 0.3216, 0.2963, 0.2984, 0.3049, 0.3206, 0.
 ↪2997, 0.3198, 0.3061, 0.2960, 0.2933, 0.3016, 0.2906]
alexnet_val_loss = [0.5686, 0.4750, 0.4156, 0.3906, 0.3646, 0.3618, 0.3471, 0.
 ↪3439, 0.3889, 0.3477, 0.3169, 0.3530, 0.3054, 0.3029, 0.3068, 0.2830, 0.
 ↪2825, 0.2929, 0.2852, 0.2849, 0.2666, 0.2709, 0.2678, 0.2827, 0.2717, 0.
 ↪2885, 0.4008, 0.2636, 0.2656, 0.2707, 0.2713, 0.2648, 0.2625, 0.2752, 0.
 ↪2643, 0.2543, 0.2650, 0.2626, 0.2530, 0.2518, 0.2683, 0.2561, 0.2477, 0.
 ↪2578, 0.2484, 0.2462, 0.2380, 0.2403, 0.2413, 0.2372]
alexnet_train_acc = [0.6339, 0.7431, 0.7684, 0.7810, 0.7776, 0.8029, 0.8092, 0.
 ↪7891, 0.8086, 0.8098, 0.8057, 0.8155, 0.8098, 0.8236, 0.8057, 0.8046, 0.
 ↪8253, 0.8264, 0.8328, 0.8443, 0.8368, 0.8287, 0.8276, 0.8322, 0.8195, 0.
 ↪8414, 0.8506, 0.8454, 0.8402, 0.8460, 0.8534, 0.8466, 0.8489, 0.8483, 0.
 ↪8316, 0.8466, 0.8621, 0.8517, 0.8489, 0.8460, 0.8603, 0.8598, 0.8523, 0.
 ↪8460, 0.8598, 0.8483, 0.8575, 0.8598, 0.8580, 0.8592, 0.8672]
alexnet_val_acc = [0.7540, 0.7908, 0.8483, 0.8437, 0.8598, 0.8092, 0.8322, 0.
 ↪8621, 0.8299, 0.7977, 0.8552, 0.8437, 0.8598, 0.8736, 0.8690, 0.8621, 0.
 ↪8782, 0.8690, 0.8736, 0.8736, 0.8874, 0.8920, 0.8966, 0.8713, 0.8782, 0.
 ↪8644, 0.8437, 0.8851, 0.8782, 0.8897, 0.8828, 0.8920, 0.8897, 0.8851, 0.
 ↪8943, 0.8920, 0.8874, 0.8897, 0.8966, 0.8920, 0.8782, 0.8874, 0.8874, 0.
 ↪8736, 0.8874, 0.8943, 0.8874, 0.8966, 0.8943, 0.8943, 0.8897]

# Plotting the training and validation loss for AlexNet
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(range(1, len(alexnet_train_loss) + 1), alexnet_train_loss,␣
 ↪label='Training Loss')
plt.plot(range(1, len(alexnet_val_loss) + 1), alexnet_val_loss,␣
 ↪label='Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
```

```
plt.title('AlexNet: Training and Validation Loss')
plt.legend()

# Plotting the training and validation accuracy for AlexNet
plt.subplot(1, 2, 2)
plt.plot(range(1, len(alexnet_train_acc) + 1), alexnet_train_acc, label='Training Accuracy')
plt.plot(range(1, len(alexnet_val_acc) + 1), alexnet_val_acc, label='Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.title('AlexNet: Training and Validation Accuracy')
plt.legend()

# Show plot
plt.tight_layout()
plt.show()
```

[3]:
```
import matplotlib.pyplot as plt

# Define the training and validation loss and accuracy values for ResNet-50
resnet_train_loss = [0.8500, 0.5180, 0.4154, 0.4130, 0.3464, 0.3427, 0.3083, 0.3229, 0.3008, 0.2965, 0.2880, 0.2900, 0.2908, 0.2650, 0.2995, 0.2951, 0.2876, 0.3078, 0.2956, 0.3032, 0.2894, 0.2736, 0.2726, 0.3106, 0.2931, 0.2706, 0.2879, 0.2919, 0.2761, 0.2899, 0.2970, 0.3027, 0.2952, 0.2989, 0.2939, 0.2983, 0.2948, 0.2694, 0.2865, 0.2949]
```

```python
resnet_val_loss = [0.9071, 0.5162, 0.3825, 0.3478, 0.2900, 0.3329, 0.2331, 0.2488, 0.3104, 0.2231, 0.2130, 0.2198, 0.2100, 0.2118, 0.2094, 0.2048, 0.2127, 0.2053, 0.2044, 0.2173, 0.2091, 0.2096, 0.2122, 0.2081, 0.2013, 0.2127, 0.2165, 0.2161, 0.2024, 0.2069, 0.2102, 0.2216, 0.2078, 0.2142, 0.2115, 0.2084, 0.2134, 0.2148, 0.2074, 0.2104]
resnet_train_acc = [0.5920, 0.7713, 0.8201, 0.8098, 0.8402, 0.8397, 0.8552, 0.8511, 0.8718, 0.8626, 0.8632, 0.8644, 0.8770, 0.8741, 0.8609, 0.8678, 0.8747, 0.8546, 0.8603, 0.8690, 0.8770, 0.8810, 0.8810, 0.8609, 0.8667, 0.8764, 0.8701, 0.8730, 0.8718, 0.8678, 0.8603, 0.8580, 0.8655, 0.8632, 0.8649, 0.8724, 0.8707, 0.8805, 0.8649, 0.8695]
resnet_val_acc = [0.6667, 0.7793, 0.8161, 0.8437, 0.8713, 0.8414, 0.9034, 0.8805, 0.8598, 0.8966, 0.8897, 0.8851, 0.8897, 0.8851, 0.8851, 0.8897, 0.8874, 0.8897, 0.8851, 0.8943, 0.8874, 0.8920, 0.8874, 0.8897, 0.8897, 0.8897, 0.8805, 0.8897, 0.8920, 0.8920, 0.8943, 0.8920, 0.8920, 0.8897, 0.8897, 0.8989, 0.8920, 0.8851, 0.8943, 0.8874]

# Plotting the training and validation loss for ResNet-50
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(range(1, len(resnet_train_loss) + 1), resnet_train_loss, label='Training Loss')
plt.plot(range(1, len(resnet_val_loss) + 1), resnet_val_loss, label='Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.title('ResNet-50: Training and Validation Loss')
plt.legend()

# Plotting the training and validation accuracy for ResNet-50
plt.subplot(1, 2, 2)
plt.plot(range(1, len(resnet_train_acc) + 1), resnet_train_acc, label='Training Accuracy')
plt.plot(range(1, len(resnet_val_acc) + 1), resnet_val_acc, label='Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.title('ResNet-50: Training and Validation Accuracy')
plt.legend()

# Show plot
plt.tight_layout()
plt.show()
```



[4]:
```
import matplotlib.pyplot as plt

# Training and validation loss and accuracy values
train_loss = [0.8224, 0.4892, 0.4770, 0.4391, 0.4026, 0.3576, 0.3403, 0.3737, 0.
↪3198, 0.3341,
              0.2972, 0.2951, 0.3049, 0.2869, 0.2909, 0.2783, 0.2594, 0.2880, 0.
↪2702, 0.2655,
              0.2678, 0.2543, 0.2490, 0.2490, 0.2564, 0.2692, 0.2504, 0.2406, 0.
↪2351, 0.2320,
              0.2175, 0.2133, 0.2070, 0.2313, 0.2220, 0.2239, 0.2204, 0.2165, 0.
↪2100, 0.2166,
              0.2161, 0.2194, 0.2201, 0.1920, 0.2070, 0.1842]

val_loss = [0.5420, 0.3861, 0.3611, 0.2814, 0.3072, 0.2201, 0.1956, 0.1891, 0.
↪2057, 0.1669,
            0.1739, 0.1902, 0.2012, 0.1504, 0.1459, 0.1594, 0.1642, 0.1501, 0.
↪1595, 0.1518,
            0.1376, 0.1369, 0.1802, 0.1598, 0.1498, 0.1324, 0.1718, 0.1271, 0.
↪1411, 0.1475,
            0.1513, 0.1912, 0.2224, 0.1732, 0.1177, 0.1243, 0.4222, 0.1705, 0.
↪1204, 0.1754,
            0.1395, 0.1336, 0.2188, 0.1318, 0.2271, 0.1898, 0.1574, 0.1612, 0.
↪2483, 0.1191]

train_acc = [0.6160, 0.8020, 0.7933, 0.8040, 0.8127, 0.8400, 0.8580, 0.8380, 0.
↪8600, 0.8480,
             0.8753, 0.8740, 0.8687, 0.8707, 0.8707, 0.8727, 0.8847, 0.8740, 0.
↪8747, 0.8793,
```

```
             0.8853, 0.8920, 0.8907, 0.8893, 0.8900, 0.8793, 0.8860, 0.8920, 0.
↪9040, 0.8913,
             0.9007, 0.9080, 0.9073, 0.8980, 0.8993, 0.9047, 0.9053, 0.9047, 0.
↪9080, 0.9060,
             0.9027, 0.9013, 0.8947, 0.9160, 0.9100, 0.9280, 0.9160, 0.9100, 0.
↪9280]

val_acc = [0.7760, 0.8560, 0.8640, 0.8987, 0.8533, 0.9013, 0.9360, 0.9253, 0.
↪9147, 0.9360,
           0.9387, 0.9253, 0.9147, 0.9413, 0.9440, 0.9493, 0.9413, 0.9387, 0.
↪9387, 0.9387,
           0.9547, 0.9387, 0.9173, 0.9360, 0.9493, 0.9493, 0.9360, 0.9440, 0.
↪9413, 0.9440,
           0.9520, 0.9253, 0.8853, 0.9280, 0.9520, 0.9467, 0.8613, 0.9227, 0.
↪9547, 0.9227,
           0.9360, 0.9307, 0.9147, 0.9413, 0.9200, 0.9280, 0.9253, 0.9467, 0.
↪9227, 0.9387]

epochs = range(1, len(train_loss) + 1)

# Plotting training and validation loss
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.plot(epochs, train_loss[:len(epochs)], 'b', label='Training loss')
plt.plot(epochs, val_loss[:len(epochs)], 'r', label='Validation loss')
plt.title('ResNet-18: Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

# Plotting training and validation accuracy
plt.subplot(1, 2, 2)
plt.plot(epochs, train_acc[:len(epochs)], 'b', label='Training accuracy')
plt.plot(epochs, val_acc[:len(epochs)], 'r', label='Validation accuracy')
plt.title('ResNet-18: Training and Validation Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.tight_layout()
plt.show()
```


ResNet-18: Training and Validation Loss
Training loss
Validation loss
Loss
Epochs
ResNet-18: Training and Validation Accuracy
Training accuracy
Validation accuracy
Accuracy
Epochs