Students: Blanca Cabrera Gil & Laila Niazy
Course: CM2003 Deep Learning Methods for Medical Image Analysis

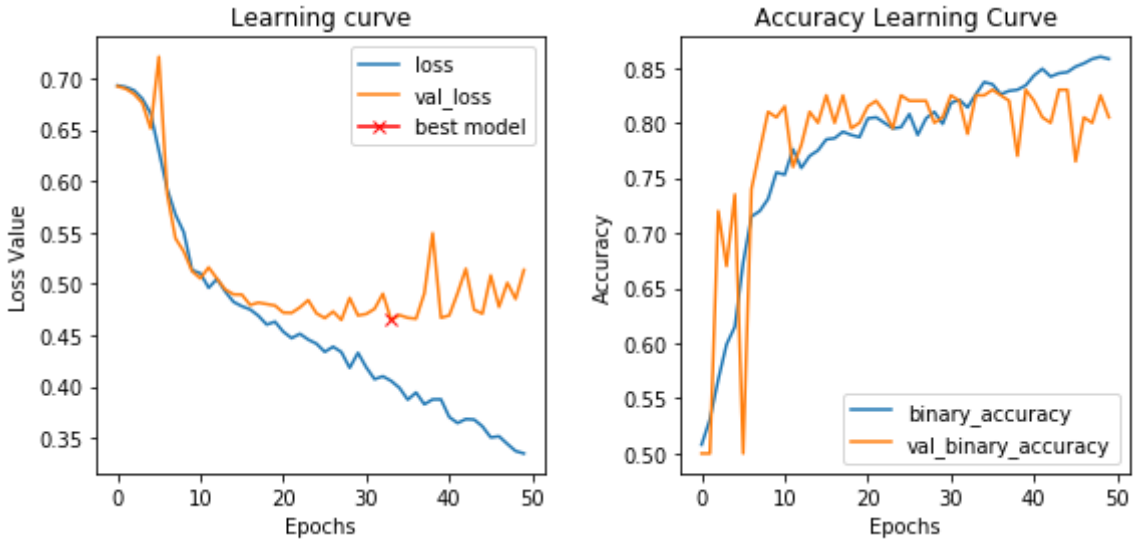
LAB 2

Regularization Techniques

Task 1a)

Employ the AlexNet model with the following architecture: five convolutional layers (base=8), followed by three dense layers (64,64,1), three max-pooling layers after 1st, 2nd, and 5th blocks; LR=0.0001, batch size=8,Adam optimizer, and image size=(128,128,1). Train this model for 50 epochs on skin images. What is the final training accuracy?

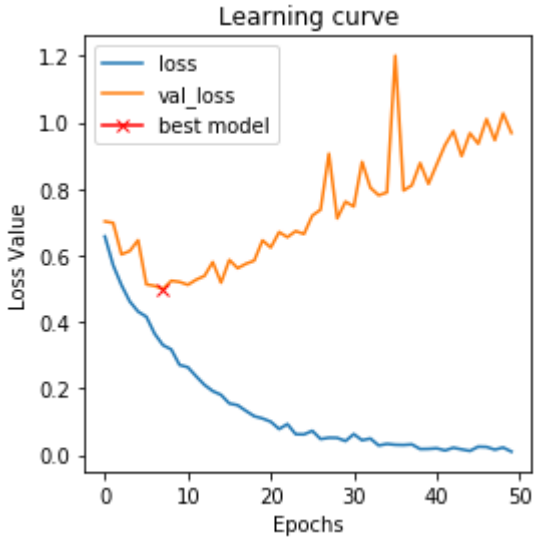
Final training accuracy: 0.8580

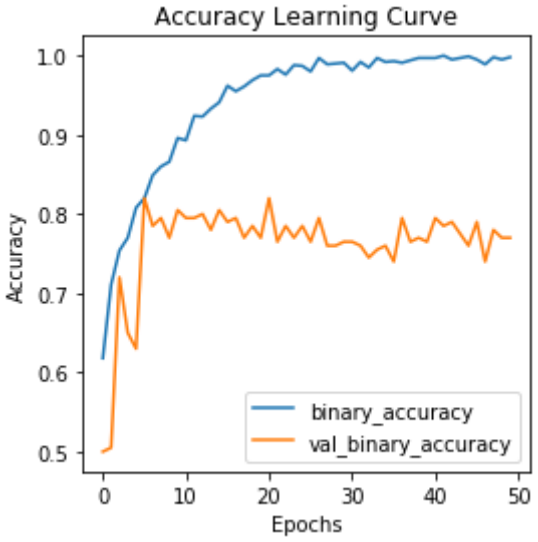


Task 1b)

With the same model and same settings, now , insert a batch normalization layer at each convolutional blocks (right after convolution layer and before activation function). At which epoch do you observe the same training accuracy as task 1a? What is the final training accuracy? What is the effect of batch normalization layer?

Same training accuracy as Task1a was reached at the 8th epoch.
Final training accuracy: 0.9980
The effect of the batch normalization layers is stabilize the learning process and reducing the number of training epochs required. It basically normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation. It is very useful for deep architectures.



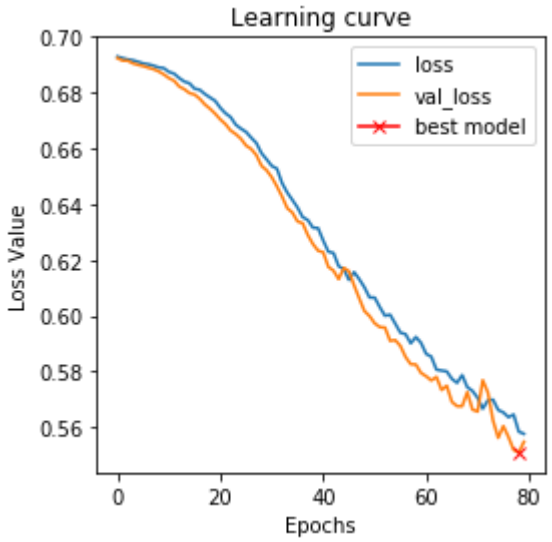


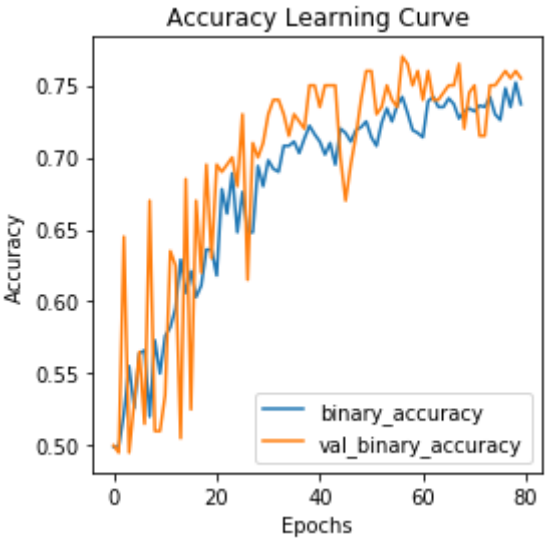
Task 1c)

Train again the same model with exact same parameters except LR = 0.00001 and epochs = 80 with and without batch normalization layers. Focus on validation accuracy. Which model resulted in higher validation accuracy? Which model resulted in higher generalization power? How do you justify the effect of batch normalization?

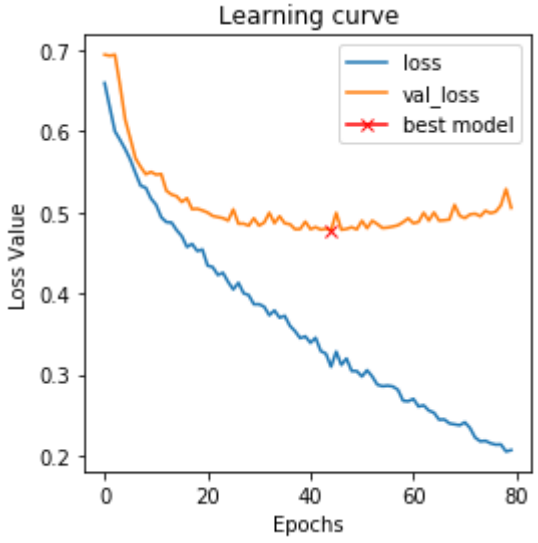
- Model resulted in higher validation accuracy: the one with BN with a val_accuracy of 0.7800 compared to a 0.7550 of the model without.
- Higher generalization power: the one with BN has a greater generalization power since it has a higher validation accuracy.
- The effect of batch normalization: As BN acts as a stabilizer of the learning, it also guarantees that the model will not get tricked by wrong data samples.

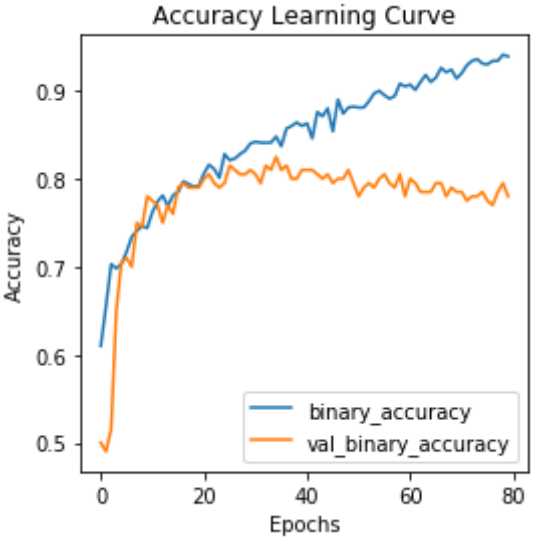
Model without BN learning curves:





Model with BN learning curves:



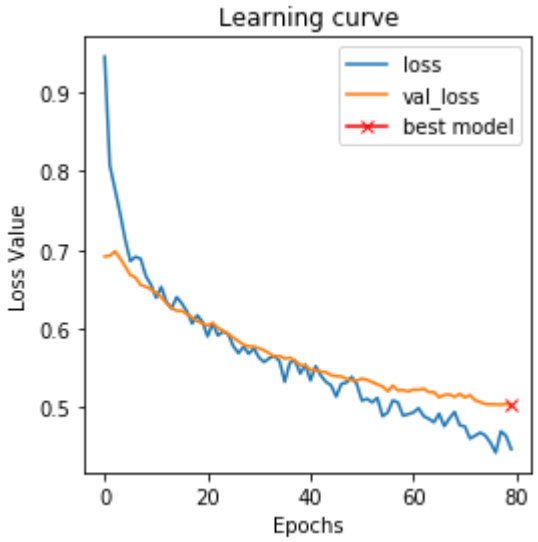


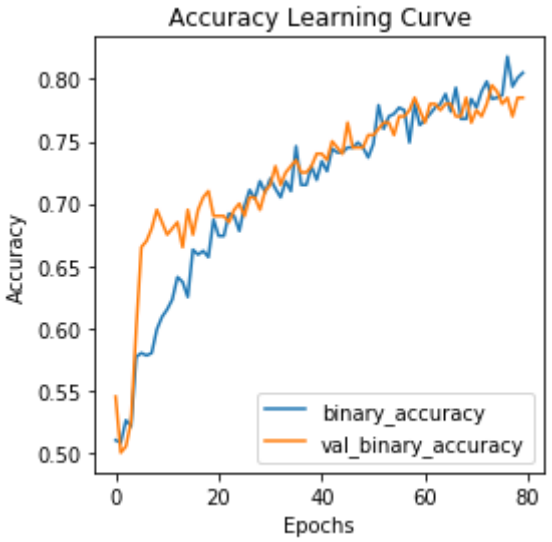
Task 2)

Use the same model as task 1c with the exact same parameters. Add drop out layers after the first two fully connected layers (right after activation layers with drop out rate of 0.4). Run the model with/without batch normalization layers and discuss the observed results. How does dropout layer help your model?

BN model:

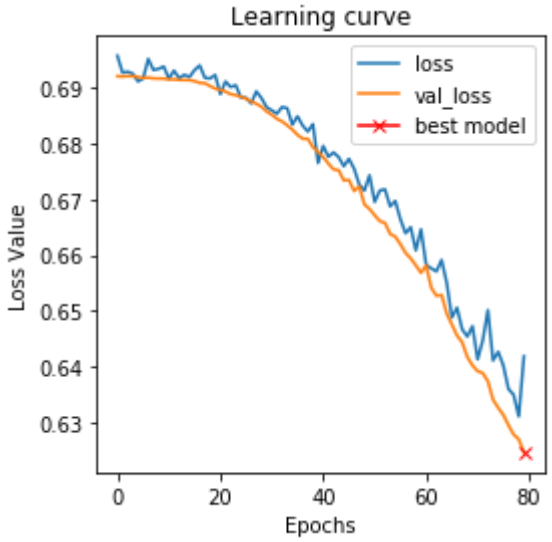
Train accuracy: 0.8050 Validation accuracy: 0.7850

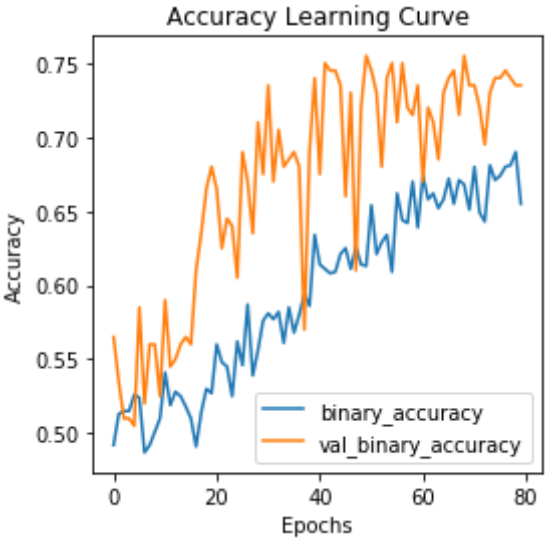




Not-BN model:

Train accuracy: 0.655 Validation accuracy: 0.7350





Conclusion: Dropout increases the regularization of the model. As can be observed in the graphs, the learning is much more stable and less fluctuating. Also, it increases the generalization power of the network. On the other hand, the effect of BN is that the learning is quicker as the model is more robust when facing misaligned data samples.

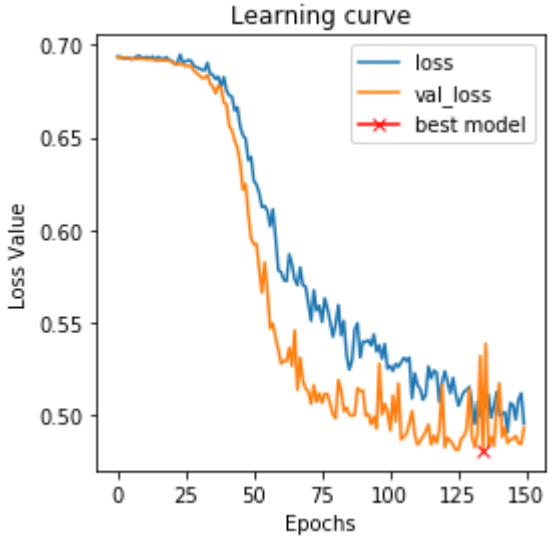
Task 3)

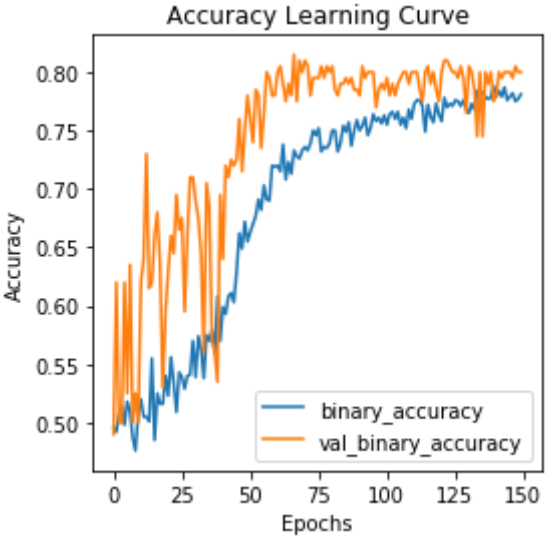
Use the same model as task 1c but set the Base parameters as 64 and remove the batch normalization layers. Instead, insert spatial dropout layers at each convolutional block (after activation function, and before max-pooling). Set the dropout rate of spatial dropout layers as 0.1 and the rate of 0.4 for the drop out layers after the first fully connected layers. Then let the model runs for 150 epochs with

LR=0.00001. Save the loss and accuracy values for the validation data. Then, run the same model with the same settings but remove all the dropout layers. Which models perform better? Which models resulted in higher generalization power? In general, discuss how the drop out layers would be helpful?

Model with dropout:

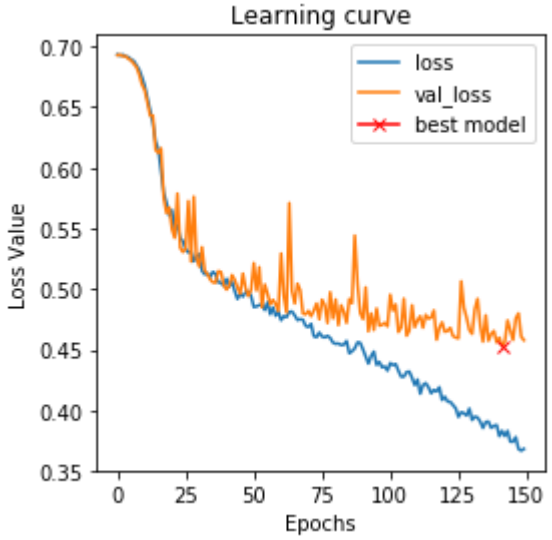
Validation loss: 0.4935 Validation accuracy: 0.8000

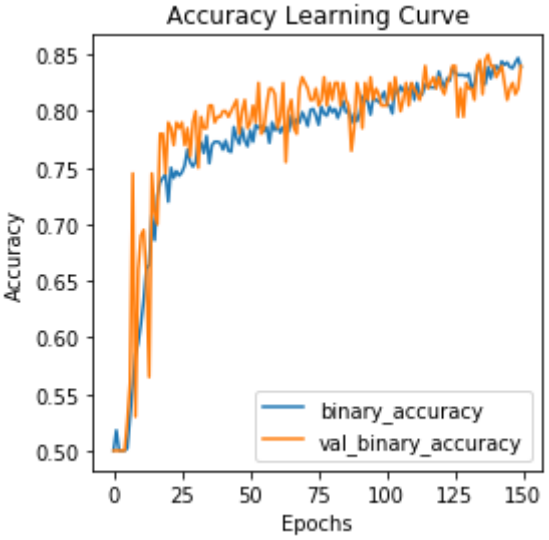




Model without dropout:

Validation loss: 0.4578 Validation accuracy: 0.8400





Conclusion: The model without dropout finished with a higher validation accuracy. However from the loss plot we can see as the training loss is decreasing at a higher rate than the validation loss. This is due to the lack of regularization in this model. When observing the

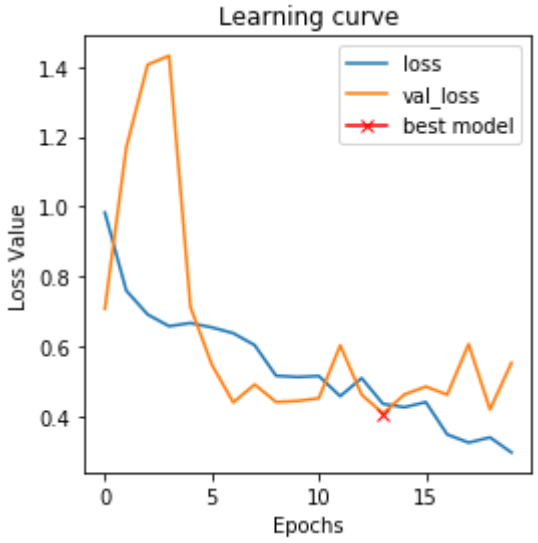
plots of the model with dropout it can be observed that the loss and accuracy curves follow the same pace. Therefore, the model with dropout will have more generalization power than the one without dropout.

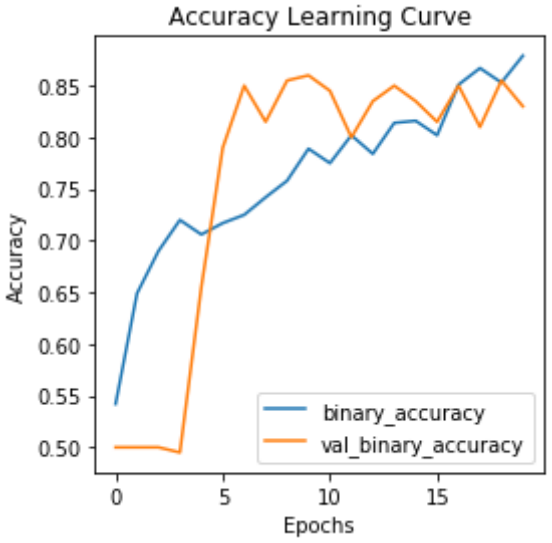
Task 4)

Try to improve the performance of VGG16 model while prevent it from overfitting by using batch normalization, spatial dropout, and/or drop out techniques. Tune the model parameters to achieve the best classification accuracy over the validation set and save the observed results for both skin and bone data.

Best models achieved:

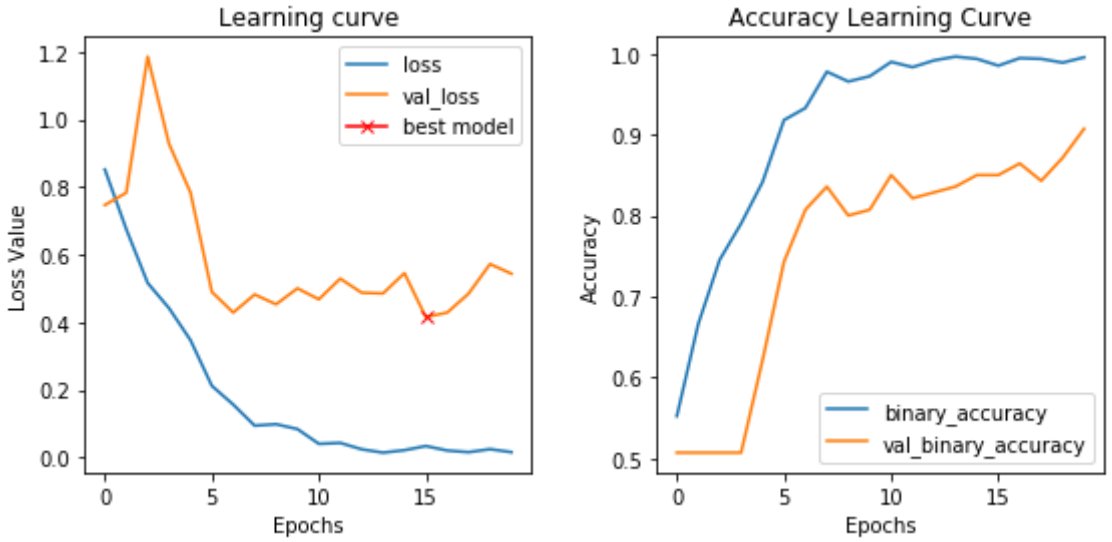
- *Skin images:*
  Training accuracy: 0.8790 Test accuracy: 0.8300





The model used for skin images contained batch normalization, spatial dropout with a rate of 0.05 and dense layer dropout with rate 0.55. It can be inferred that the model is still in the learning process. It cannot be said that the model overfits the training data as both data sets (test and training) evolve with a very similar rate.

- *Bone images:*
  Training accuracy: 0.9953 Test accuracy: 0.9071



The model that gave the best performance given the network parameters for VGG contained batch normalization and a dropout layers after the dense layers of 0.4. However, when testing with the spatial dropout resulted in similar accuracy evolution of the validation and training set. Yet, for the number of epochs the final accuracy was lower.

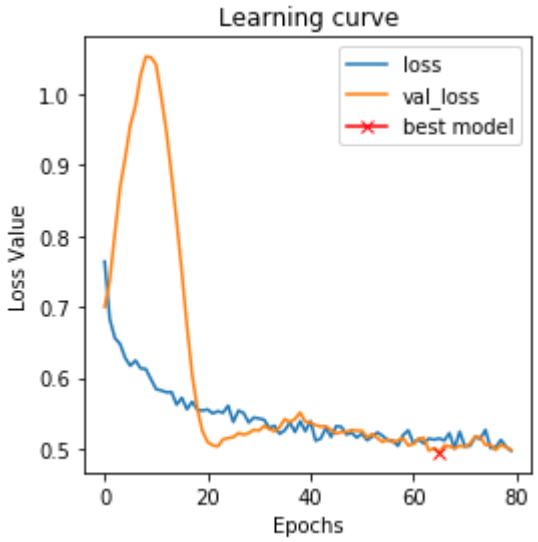
Data Augmentation

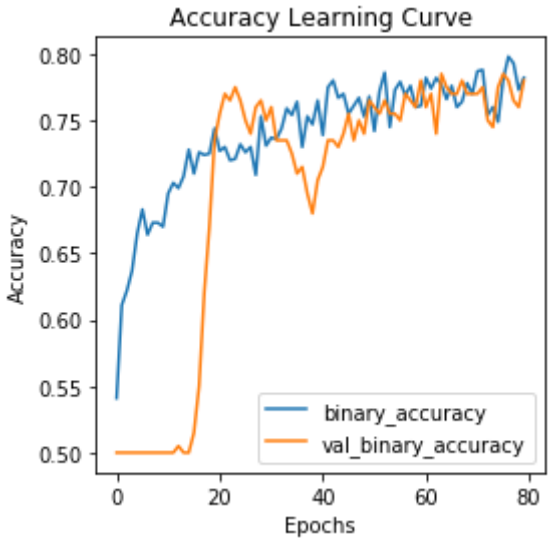
Task 6)

Does data augmentation help to improve the model performance? Why?

Yes, data augmentation helps to improve the model performance and generalization power. This is due to the fact that the creation of new data and new images provides the network with more knowledge, containing samples different from each other. This way, the model can learn features that were not previously known in the dataset and perform better when facing unseen data.

Training accuracy: 0.7820 Validation accuracy: 0.7800



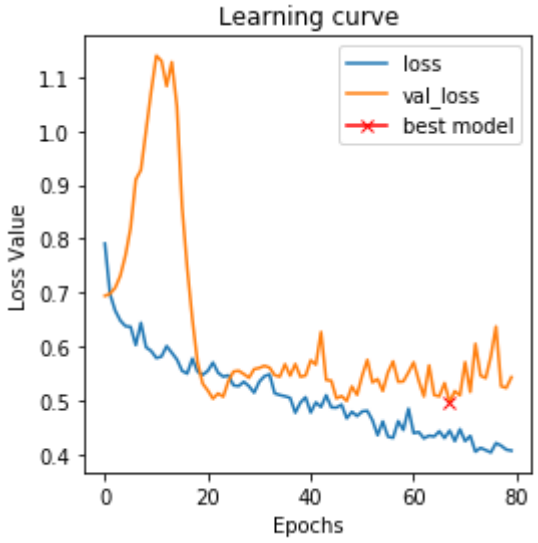


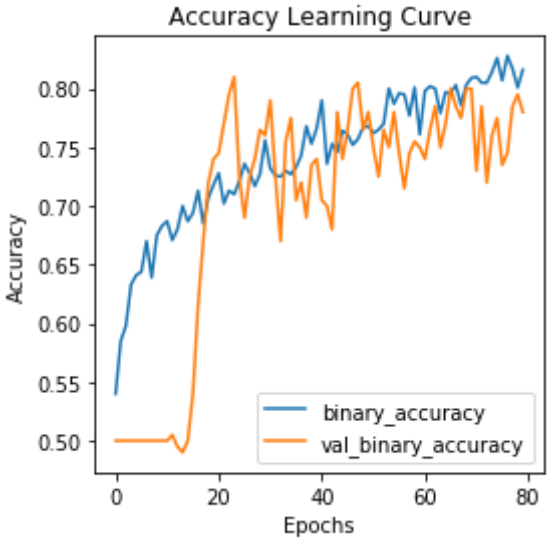
Task 7)

Repeat task 6 for VGG model for both skin and bone data set.

Skin images results:

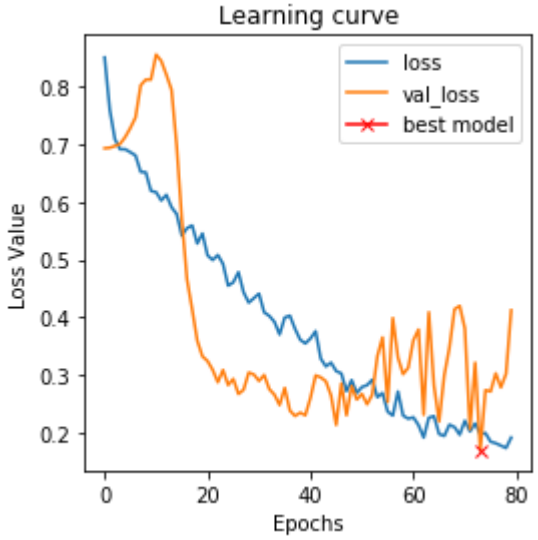
Training accuracy: 0.8160 Test accuracy: 0.7800

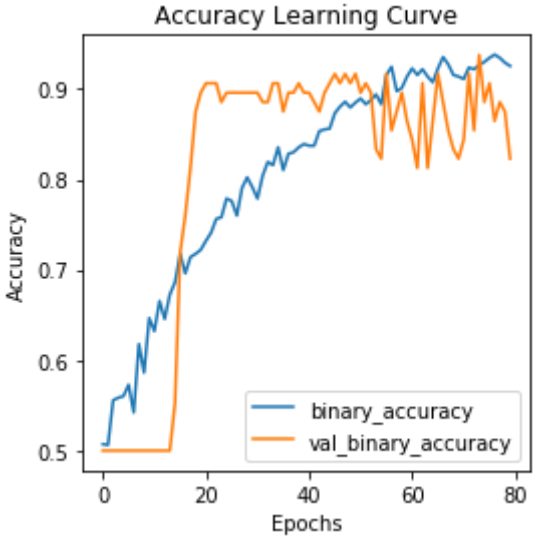




Bone results:

Training accuracy: 0.9379 Test accuracy: 0.9375





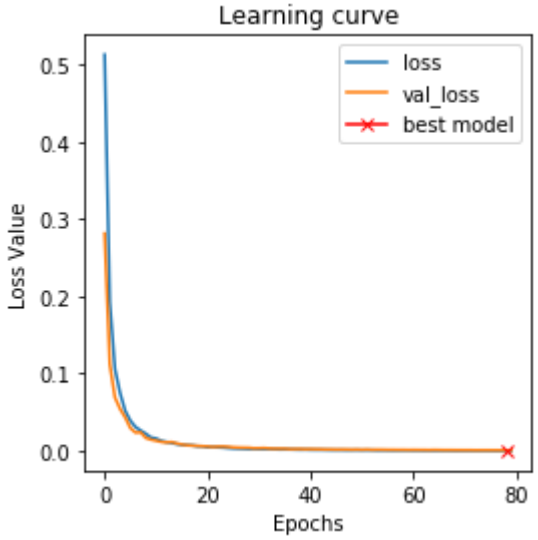
Transfer Learning

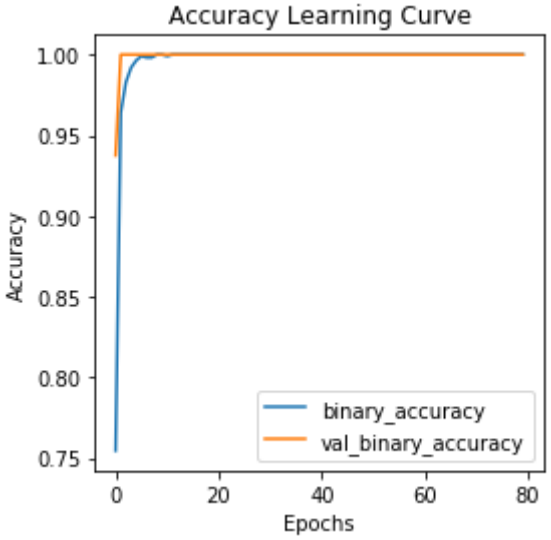
Task 9)

Train the fine-tuned model with skin and bone images. Do you observe some changes in model performance? Describe how transfer learning would be useful for training? How can you make sure that the results are reliable?

Bone results:

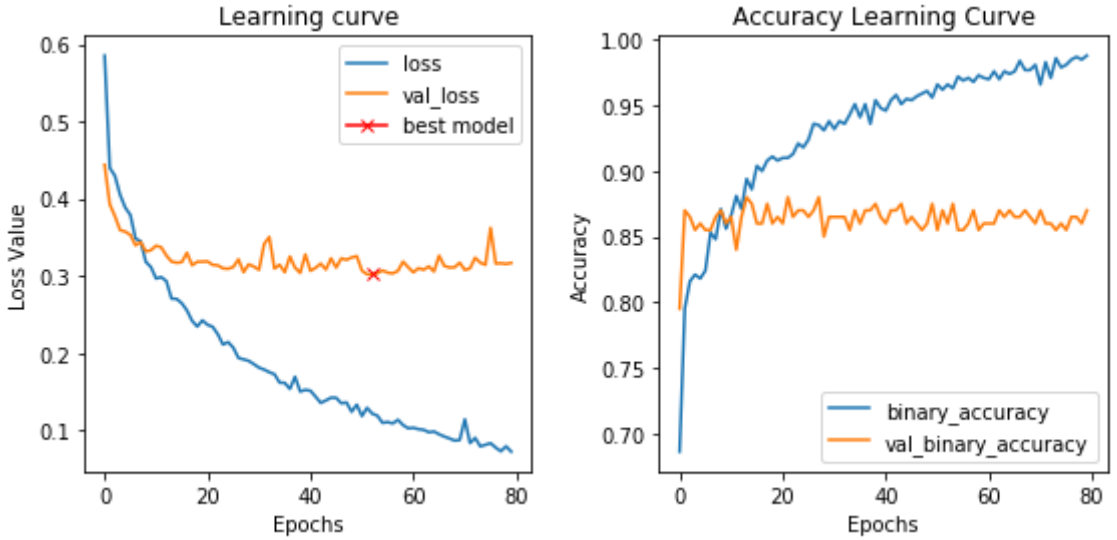
Training accuracy: 1.00 Test accuracy: 1.00





Skin results:

Training accuracy: 0.9880 Test accuracy: 0.8700



Conclusion:

The model has a way better performance on the training and validation set. Transfer learning can be useful in decreasing the time needed for training**.** Further, the rate of improvement of skill during training of the source model is steeper than it otherwise would be. The converged skill of the trained model is better than it otherwise would be.

Task 10)

Back to bone image classification. Design a VGG16 with 2 neurons at the last year so that the activation function should be set as "softmax" and categorical cross entropy as loss function. (Remember to name the last convolutional layer as name = 'Last_ConvLayer'). Train the model with data augmentation techniques and after the model learned, follow the implementation below and interpret the observed results.