CNN for classification of skin lesions using the HAM10000 dataset

Dataset
The HAM10000 dataset contains 10015 dermatoscopic images of skin lesions labeled with one of seven diagnostic categories: melanoma, melanocytic nevus, basal cell carcinoma, actinic keratosis, benign keratosis, dermatofibroma, and vascular lesion. The images have different sizes and resolutions, and are imbalanced in terms of class distribution. Therefore, some preprocessing steps are needed before feeding them to the CNN model.

Data Preprocessing
The preprocessing steps include:
- Resizing all images to a fixed size of 90x120 pixels, which is a trade-off between preserving the details of the lesions and reducing the computational cost.
- Normalizing the pixel values to the range [0, 1] by dividing them by 255.
- Splitting the dataset into training (80%), validation (10%), and test (10%) sets, and ensuring that each set has a balanced class distribution by applying stratified sampling.

CNN Architecture
The CNN model consists of several convolutional, pooling, normalization, dropout, and dense layers. The architecture is as follows:

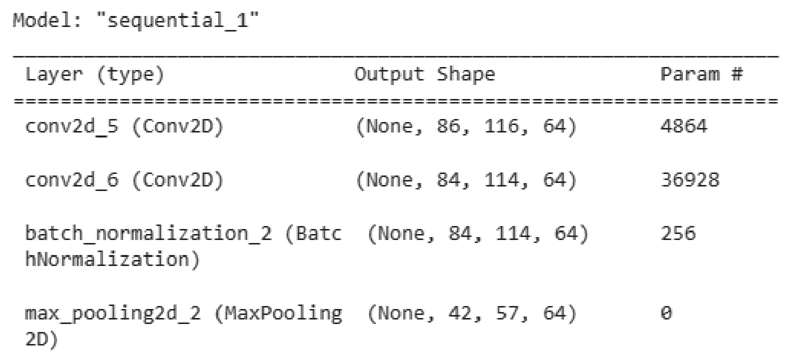
1. The first layer is a convolutional layer with 64 filters of size 5x5, a ReLU activation function, and an input shape of 90x120x3. This layer extracts low-level features from the input images, such as edges and corners.
2. The second layer is another convolutional layer with 64 filters of size 3x3 and a ReLU activation function. This layer further refines the features extracted by the previous layer.
3. The third layer is a batch normalization layer, which normalizes the outputs of the previous layer to have zero mean and unit variance. This layer helps to speed up the training process and reduce overfitting by reducing the internal covariate shift.
4. The fourth layer is a max pooling layer with a pool size of 2x2. This layer reduces the spatial dimensions of the feature maps by taking the maximum value in each 2x2 region. This layer also helps to reduce overfitting by introducing some translational invariance.
5. The fifth layer is a convolutional layer with 128 filters of size 3x3 and a ReLU activation function. This layer extracts higher-level features from the feature maps, such as shapes and patterns.
6. The sixth layer is another convolutional layer with 128 filters of size 3x3 and a ReLU activation function. This layer further refines the features extracted by the previous layer.
7. The seventh layer is another batch normalization layer, which normalizes the outputs of the previous layer to have zero mean and unit variance.
8. The eighth layer is another max pooling layer with a pool size of 2x2. This layer reduces the spatial dimensions of the feature maps by taking the maximum value in each 2x2 region.
9. The ninth layer is a convolutional layer with 256 filters of size 3x3 and a ReLU activation function. This layer extracts even higher-level features from the feature maps, such as textures and parts.

10. The tenth layer is another convolutional layer with 256 filters of size 3x3 and a ReLU activation function. This layer further refines the features extracted by the previous layer.
11. The eleventh layer is a group normalization layer, which divides the channels of the previous layer into groups and normalizes each group separately. This layer helps to improve the performance of the model when the batch size is small or when using distributed training.
12. The twelfth layer is another max pooling layer with a pool size of 2x2. This layer reduces the spatial dimensions of the feature maps by taking the maximum value in each 2x2 region.
13. The thirteenth layer is a convolutional layer with 512 filters of size 3x3 and a ReLU activation function. This layer extracts very high-level features from the feature maps, such as objects and scenes.
14. The fourteenth layer is another convolutional layer with 512 filters of size 3x3 and a ReLU activation function. This layer further refines the features extracted by the previous layer.
15. The fifteenth layer is another batch normalization layer, which normalizes the outputs of the previous layer to have zero mean and unit variance.
16. The sixteenth layer is another max pooling layer with a pool size of 2x2. This layer reduces the spatial dimensions of the feature maps by taking the maximum value in each 2x2 region.
17. The seventeenth layer is a flatten layer, which reshapes the feature maps into a one-dimensional vector. This layer prepares the input for the dense layers.
18. The eighteenth layer is a dense layer with 1024 units and a ReLU activation function. This layer performs a linear transformation on the input vector and applies a nonlinear activation function. This layer acts as a fully connected layer that learns the global features of the input.
19. The nineteenth layer is another batch normalization layer, which normalizes the outputs of the previous layer to have zero mean and unit variance.
20. The twentieth layer is a dropout layer with a rate of 0.5. This layer randomly drops out some units of the previous layer during training, which helps to prevent overfitting by reducing the co-adaptation of features.
21. The twenty-first layer is another dense layer with 512 units and a ReLU activation function. This layer performs another linear transformation on the input vector and applies a nonlinear activation function. This layer acts as another fully connected layer that learns more global features of the input.
22. The twenty-second layer is the output layer, which is a dense layer with 7 units and a softmax activation function. This layer performs the final linear transformation on the input vector and applies a softmax function, which normalizes the output to a probability distribution over the 7 classes. This layer predicts the class label of the input image.

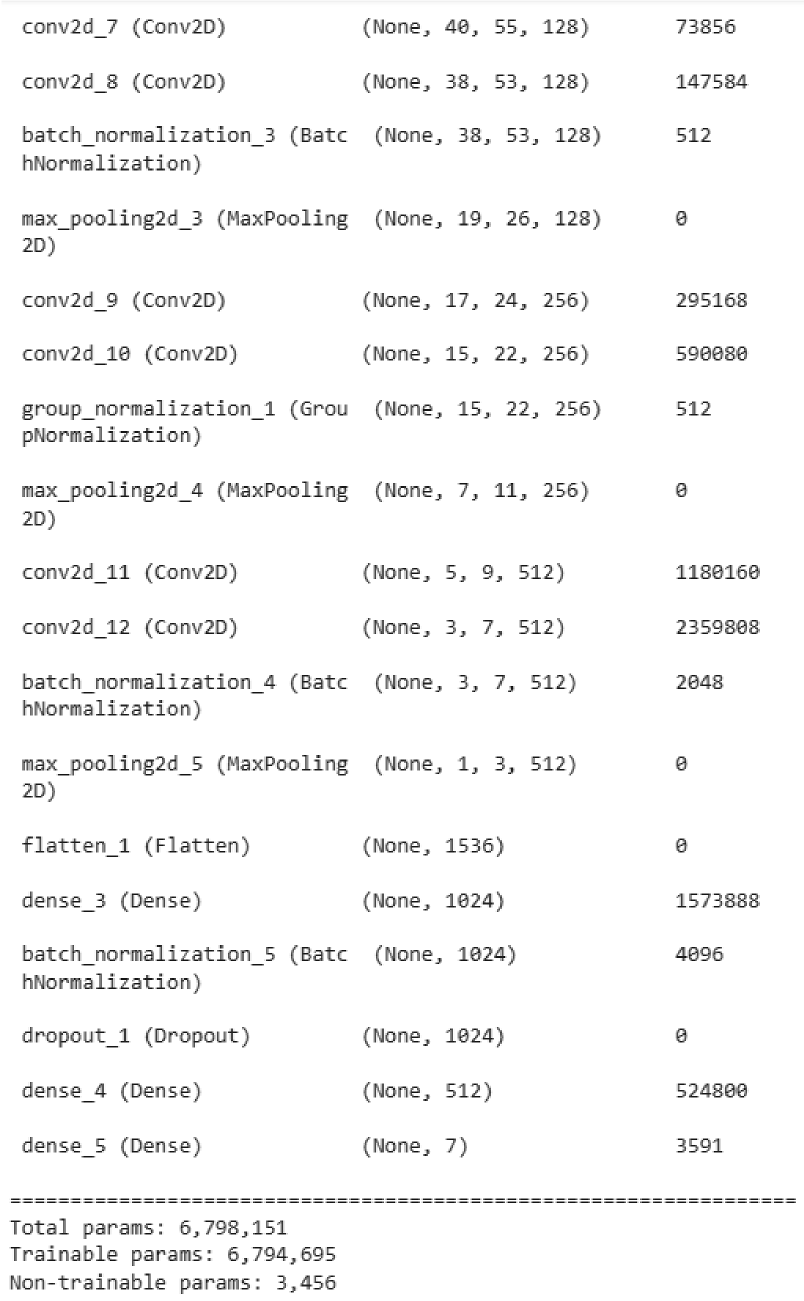
The model is compiled with the Adam optimizer, which is an adaptive learning rate algorithm that adjusts the learning rate based on the gradients. The loss function used is the categorical cross-entropy, which measures the difference between the predicted probabilities and the true labels. The metric used to evaluate the model is the accuracy, which measures the fraction of correctly classified images.

Model Sumary

The model summary shows the number of parameters, output shapes, and memory usage of each layer in the model.

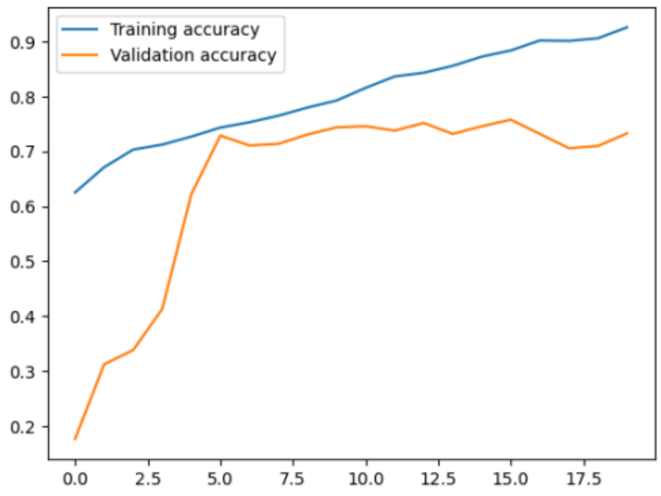


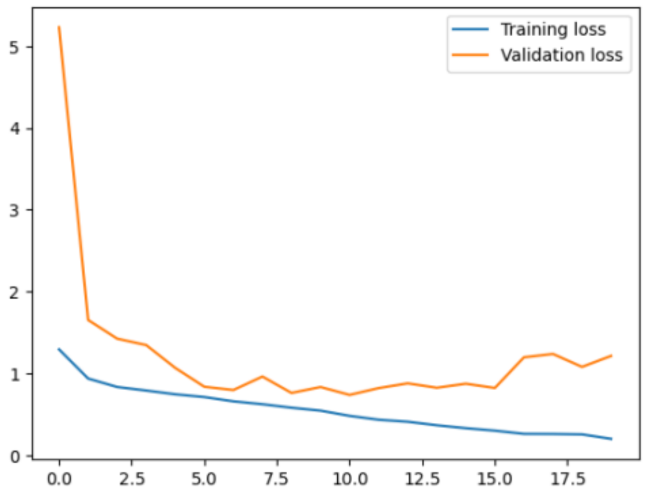
```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_5 (Conv2D)           (None, 86, 116, 64)       4864

 conv2d_6 (Conv2D)           (None, 84, 114, 64)       36928

 batch_normalization_2 (Batc  (None, 84, 114, 64)      256
 hNormalization)

 max_pooling2d_2 (MaxPooling  (None, 42, 57, 64)       0
 2D)
```



```
conv2d_7 (Conv2D)           (None, 40, 55, 128)       73856

conv2d_8 (Conv2D)           (None, 38, 53, 128)       147584

batch_normalization_3 (Batc  (None, 38, 53, 128)      512
hNormalization)

max_pooling2d_3 (MaxPooling  (None, 19, 26, 128)      0
2D)

conv2d_9 (Conv2D)           (None, 17, 24, 256)       295168

conv2d_10 (Conv2D)          (None, 15, 22, 256)       590080

group_normalization_1 (Grou  (None, 15, 22, 256)      512
pNormalization)

max_pooling2d_4 (MaxPooling  (None, 7, 11, 256)       0
2D)

conv2d_11 (Conv2D)          (None, 5, 9, 512)         1180160

conv2d_12 (Conv2D)          (None, 3, 7, 512)         2359808

batch_normalization_4 (Batc  (None, 3, 7, 512)        2048
hNormalization)

max_pooling2d_5 (MaxPooling  (None, 1, 3, 512)        0
2D)

flatten_1 (Flatten)         (None, 1536)              0

dense_3 (Dense)             (None, 1024)              1573888

batch_normalization_5 (Batc  (None, 1024)             4096
hNormalization)

dropout_1 (Dropout)         (None, 1024)              0

dense_4 (Dense)             (None, 512)               524800

dense_5 (Dense)             (None, 7)                 3591

=================================================================
Total params: 6,798,151
Trainable params: 6,794,695
Non-trainable params: 3,456
```

Result plots


Training accuracy
Validation accuracy


Training loss
Validation loss