IMAGE ANALYSIS FOR AUTOMATIC DOCUMENT DIGITIZATION

Richard Han
rickyhan24@gmail.com

ABSTRACT

In this project, I perform image and video analysis to help automate the process of document digitization. Images of book pages being flipped or not flipped are classified using a traditional convolutional neural network and a fine-tuned pretrained model Resnet50. Videos in the form of a sequence of frames are classified as flipping or not flipping using a combination of a CNN and LSTM. These analyses play a role in the capability of a mobile app, MonReader, to automatically detect and capture images of the pages of a book as they are being flipped. This can improve the ease with which users of the app can scan documents compared to traditional means of scanning documents.

1 Problem Statement

The task is to classify images of book pages as flip or not flip with high accuracy and f1score. We also want to classify a sequence of images, which constitutes a video, as flip or not flip; this can be thought of as detecting a certain kind of activity or motion.

2 Data

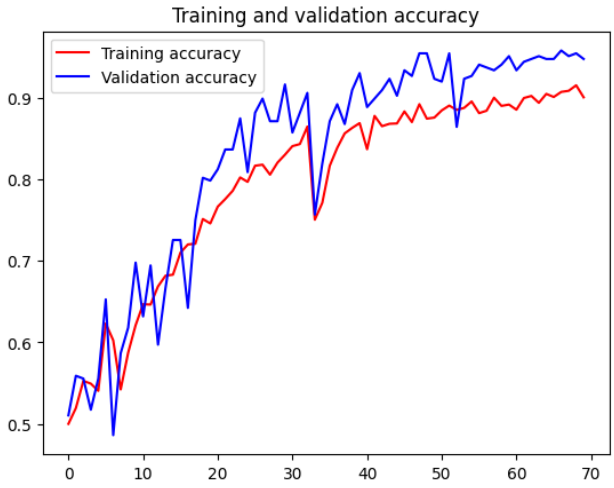
The data consists of roughly 2400 training images divided into two classes (flip and notflip), 299 validation images divided into the same two classes, and 298 test images divided into the same two classes. The images are split between flip and notflip roughly evenly in each case. The training and validation images are used to train the CNN and Resnet models, while the test images are used to assess their performance with respect to accuracy, precision, recall, and f1score.

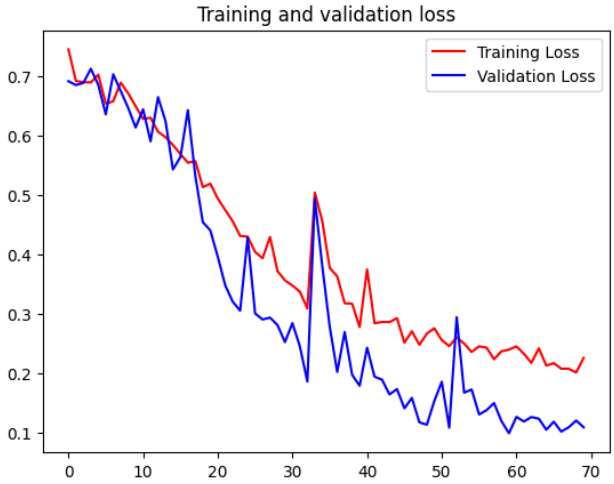
The images are labeled with the structure VideoID_FrameNumber and come in sequences of varying lengths; so a sequence of say 20 images with the same videoID form a kind of short video clip. For the sequence classification portion of this project, the images are organized so that each sequence of images that forms a short video clip gets its own folder.

3 Image Classification

3.1 CNN

I used a convolutional neural network with 3 convolutional layers, each followed by a maxpooling layer. I trained the cnn for 75 epochs and got the following plots for accuracy and loss:

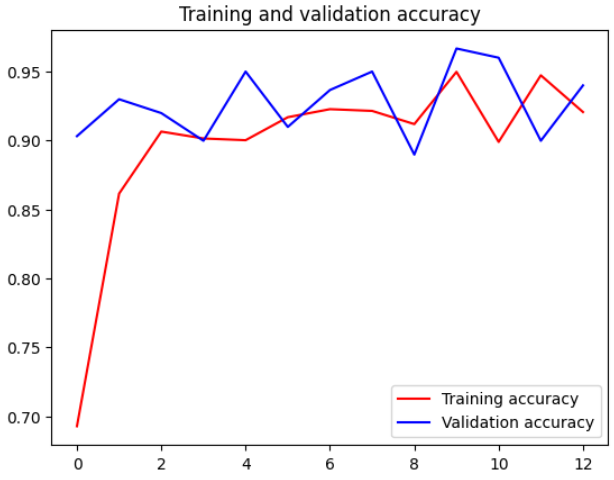


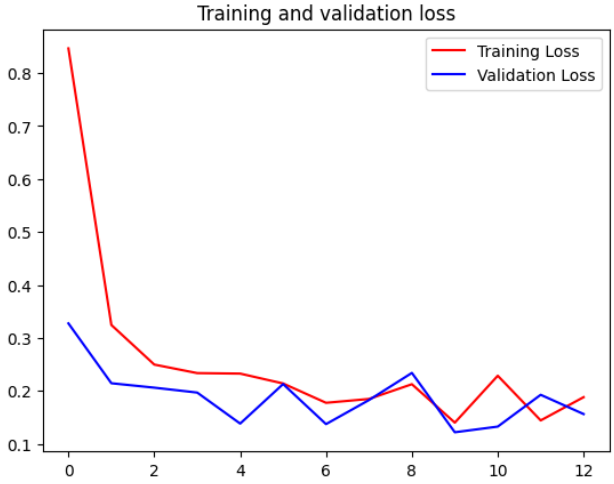


The training accuracy reached 90% and the validation accuracy reached 95%. After evaluating the model on the test set, I got a test set accuracy of 95.64%, a precision of 0.92, a recall of 1, and a f1score of 0.96.

3.2 Pretrained Model

Instead of training our own CNN, we can use a pre-trained model such as Resnet50 and use transfer learning to adapt it for our classification purposes. In this approach, I used the Resnet50 as the base model, added custom layers for the sake of binary classification, and froze the layers of the base model during training. After training with early stopping, the training ran for 13 epochs. Here are the plots for accuracy and loss:





I then fine-tuned the model by unfreezing the last 20 layers of the base model and running 5 epochs. After evaluating the model on the test set, I got a test accuracy of 99.33%, a precision of 0.99, a recall of 0.99, and a f1score of 0.99. Compared to training our own CNN model, using a pre-trained model took less epochs to train and gave better results.

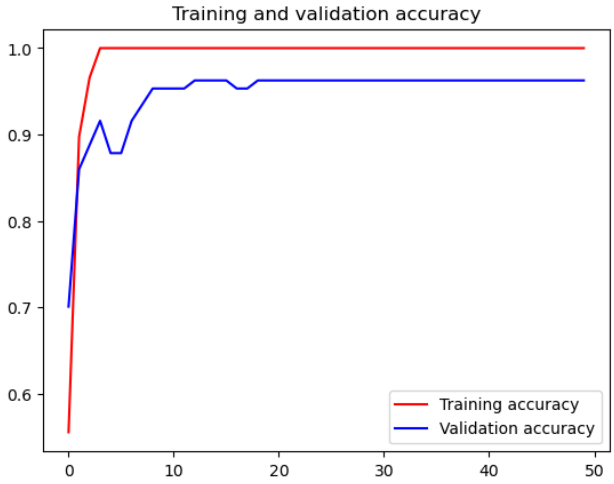
I also wanted to test my CNN and resnet models on new unseen images different from the dataset. The fine-tuned resnet model appeared to perform better than the cnn model on realistic-looking photos of pages of books; however, despite both models having high test set accuracy, they both performed poorly when it came to images that were like stock photos and not as realistic-looking images of photos of pages of books. Both models don't seem to generalize well beyond the types of images they were trained on.

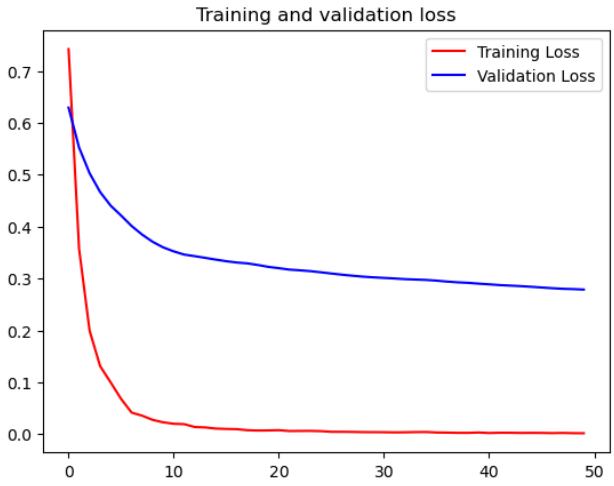
4 Sequence Classification

One of our tasks was to classify sequences of images as flip or notflip; that is, we want to classify a short video clip as displaying the flipping action or not. In order to do this, we use the Resnet50 model to extract features from each image in a given sequence and then use an LSTM to learn the sequential nature from frame to frame.

I padded the training sequences using a maxlen of 10. 95% of the training sequences have a length that is less than or equal to 27. Further, 95% of the validation sequences have a length that is less than or equal to 5. Using a maxlen of 10 is a sort of compromise and leads to good performance. I padded the validation sequences using a maxlen of 5.

After training for 50 epochs, here are the accuracy and loss plots:





Using a maxlen of 10, which is much smaller than maxlen of 27, for training and a maxlen of 5 for validation works really well for validation accuracy. You get around 96% val accuracy. Increasing maxlen for training beyond 10 doesn't seem to improve val accuracy.

To evaluate the model on the test set, I used a maxlen of 4; 95% of the test sequences have a length that is less than or equal to 6, but using a maxlen of 4 gives better results.

After evaluating on the test set, I got a test accuracy of 98.06%, a precision of 1, a recall of 0.96, and a f1score of 0.98.

5 Conclusion

In this project, I trained a traditional CNN model to classify images as flip or notflip with 95% accuracy, and I used transfer learning to fine-tune a pretrained Resnet50 model to classify images with 99% accuracy. The pretrained model performed better than the CNN and took less time to train. However, both models did not generalize well when applied to images that were not as realistic-looking as the ones the models were trained on. Finally, I applied a CNN/LSTM combined model to classify sequences of images, or 'videos', as flip or notflip with 98% accuracy. The choice of a good maxlen value was crucial in getting optimal performance.

The image and video analyses performed in this project are useful not only for the mobile app, MonReader, that automates document digitization but also for image classification tasks like medical image analysis and activity/motion detection tasks like automated surveillance.