Michael Loos

Whose Team Are You On?

Newspapers and websites often enhance the stories they publish with photos. This is done by sending photographers to events or obtaining photos from the Associated Press. This is particularly true for sporting events. If you're a national or regional outlet, you might be covering anywhere from several to hundreds of games in a short timeframe. It could prove very useful to have a model capable of predicting the team associated with the photo instead of relying on someone to spend time classifying the images

Through machine learning, computer vision can be used to determine what a photo contains and label the photo as belonging to a certain game or team. A convolutional neural network, also called convnet, is a type of deep-learning model used in nearly all cases of computer vision applications (François 119). Convnets can be trained to label an image within a dataset. This is one way to provide and sort large amounts of images at once.

Data Explanation

The first step in this project was obtaining photos for all 14 Big Ten teams. This was done by using the Selenium library in Python to create a WebDriver. The WebDriver opened a browser and searched Google Images by using the school and mascot plus "football" as the search term. (See below for list of teams.)

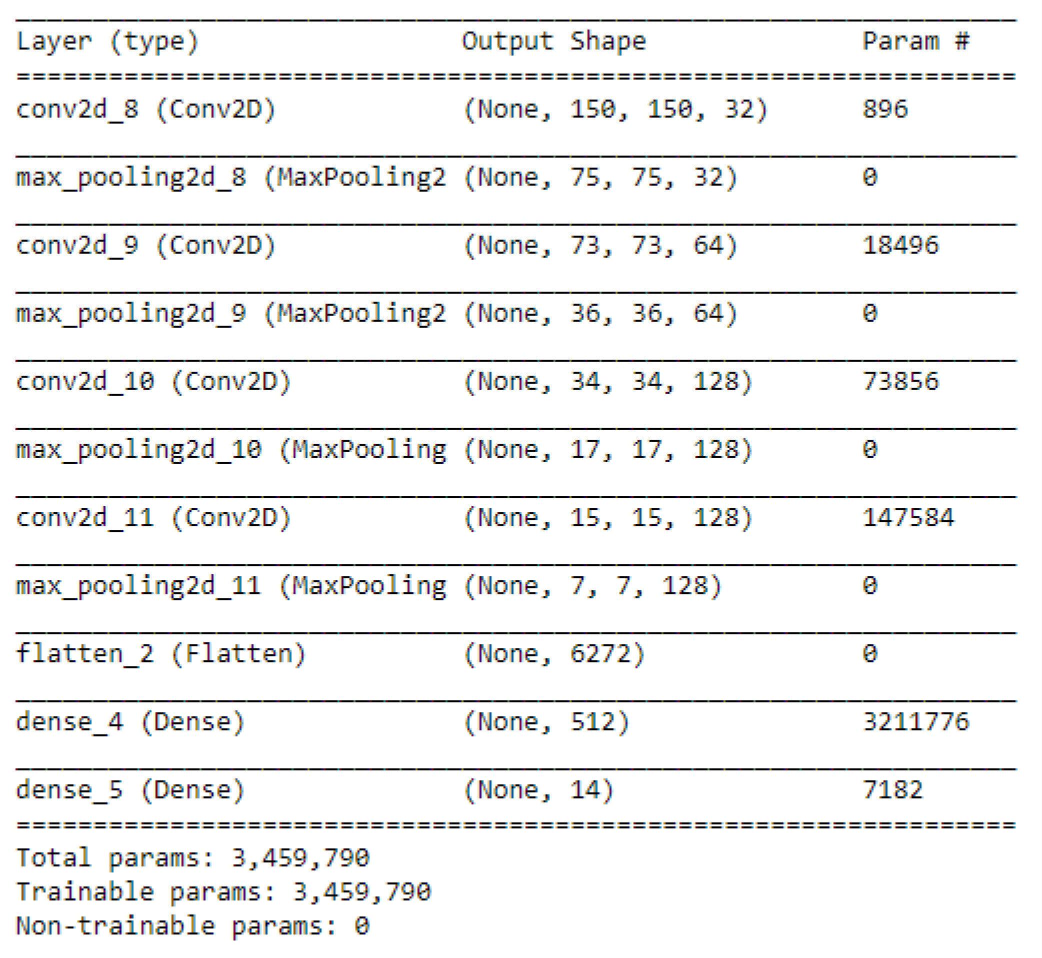
Teams: Illinois Fighting Illini, Indiana Hoosiers, Iowa Hawkeye, Maryland Terrapins, Michigan Wolverines, Michigan State Spartans, Minnesota Gophers, Nebraska Cornhuskers, Northwestern Wildcats, Ohio State Buckeyes, Penn State Nittany Lions, Purdue Boilermakers, Rutgers Scarlet Knights, Wisconsin Badgers.

Photos included any image that appeared in the search results, ranging from logos to stadiums to players. Around 500 photos were obtained for each team. 60% of photos for each term were put into a training group, 20% of photos for each team were put into a validation group and 20% of photos for each team were put into a test group.

Methods

A convolutional neural network was determined to be the best model due to a convnet's ability to work with small sample sets. The convnet contains four stages of Conv2D layers with relu activation paired with MaxPooling2D layers before reaching a Flatten layer. The network ends with two Dense layers. The final layer uses a softmax activation and has 14 outputs, one for each team. This network results in 3,459,790 total and trainable parameters. The model was compiled using categorical cross-entropy for the loss function and RMSprop(lr=1e-4) for the optimizer.

Model Summary

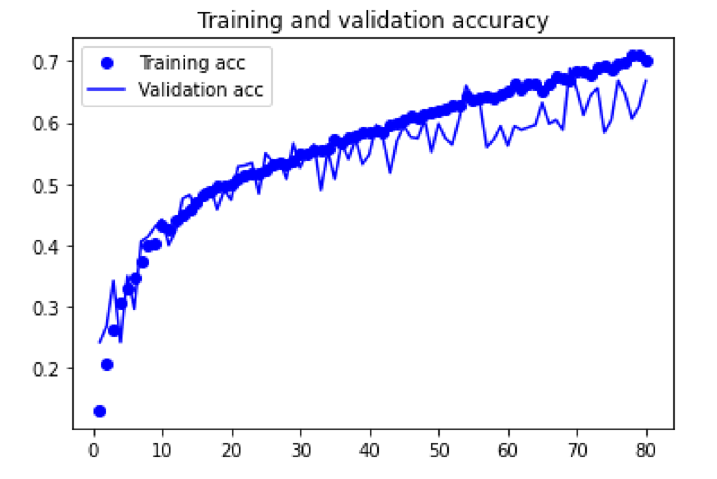


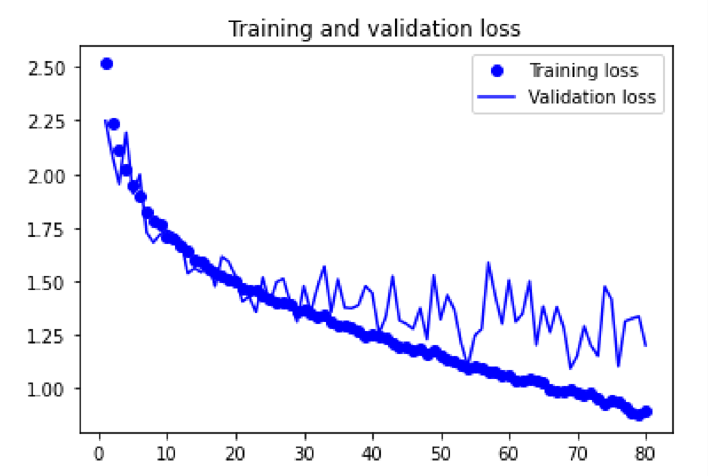
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_8 (Conv2D)            (None, 150, 150, 32)      896       
_________________________________________________________________
max_pooling2d_8 (MaxPooling2 (None, 75, 75, 32)        0         
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 73, 73, 64)        18496     
_________________________________________________________________
max_pooling2d_9 (MaxPooling2 (None, 36, 36, 64)        0         
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 34, 34, 128)       73856     
_________________________________________________________________
max_pooling2d_10 (MaxPooling (None, 17, 17, 128)       0         
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 15, 15, 128)       147584    
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 7, 7, 128)         0         
_________________________________________________________________
flatten_2 (Flatten)          (None, 6272)              0         
_________________________________________________________________
dense_4 (Dense)              (None, 512)               3211776   
_________________________________________________________________
dense_5 (Dense)              (None, 14)                7182      
=================================================================
Total params: 3,459,790
Trainable params: 3,459,790
Non-trainable params: 0
```

Generators were used to input the images into the model. Because of the relatively small sample size of images for each team, the training images were augmented in order to produce a better data set for the network. Images were rescaled, rotated, shifted vertically and horizontally, flipped, sheared, and magnified.

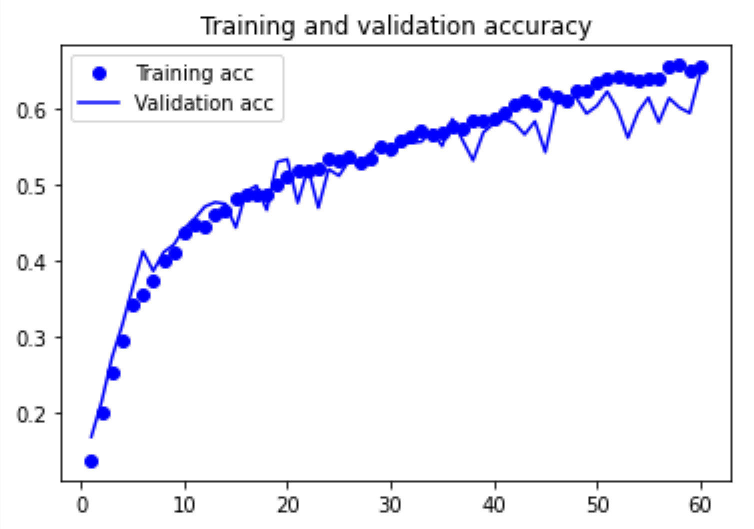
The model was trained using the input of the training generator as training data for 80 epochs with 200 steps per epoch and validation data from the validation generator with 50 validation steps.

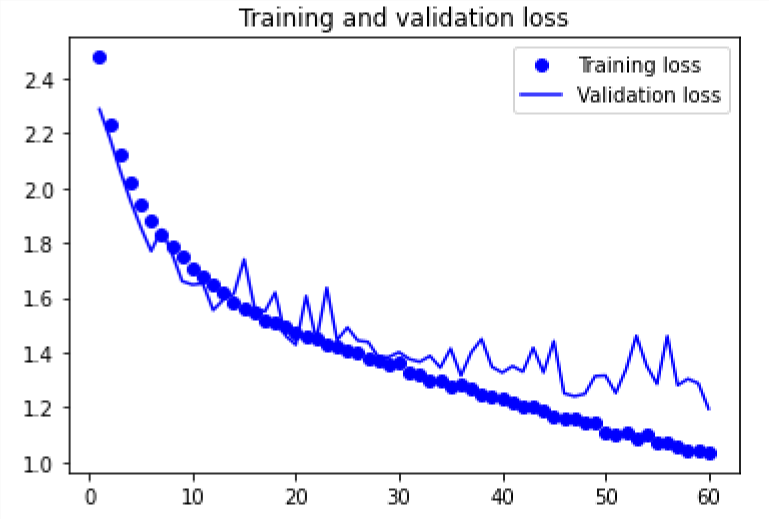
After 80 epochs, training accuracy reached 70% while validation accuracy reached 67%. There were some concerns of overfitting. If you look at graphs of accuracy and loss below, we can see the training model and validation model really start to separate around 60 epochs.





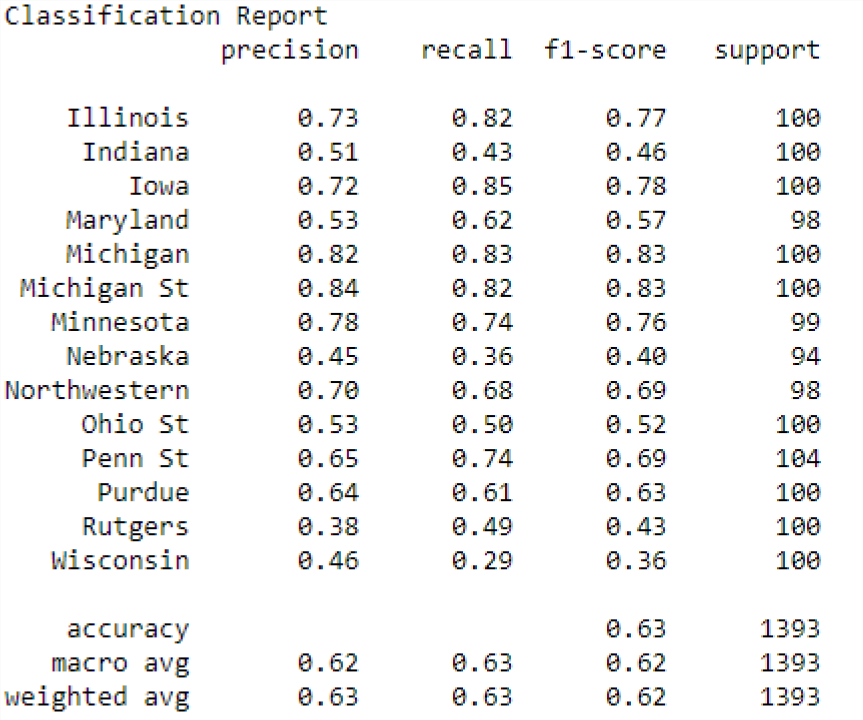
The model was retrained using 60 epochs. At 60 epochs, training accuracy reached 66% while validation accuracy reached 64%. Using the test data set, the 60-epoch model had a training accuracy of 63%. The 80-epoch model had an accuracy of 61%. While the difference in accuracy on the test data is small, the 60-epoch model was more accurate and should perform better on general data as the 80-epoch model might be overfitting.





Analysis

Running the model on the test dataset resulted in an accuracy of 63%. While not great, this is much better than the 7% (1/14 or one team out of 14) accuracy provided by a random guess. An interesting trend from the prediction data is the performance of model by team. If you look at the classification report below, we see there are a handful of teams that the model struggles with: Indiana, Maryland, Nebraska, Ohio State, Rutgers, and Wisconsin.

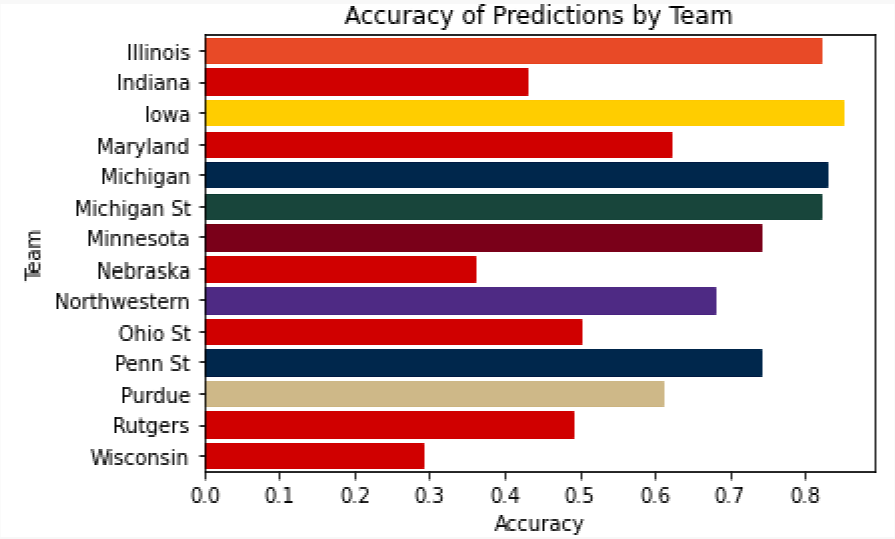


```
Classification Report
              precision    recall  f1-score   support

    Illinois       0.73      0.82      0.77       100
     Indiana       0.51      0.43      0.46       100
        Iowa       0.72      0.85      0.78       100
    Maryland       0.53      0.62      0.57        98
    Michigan       0.82      0.83      0.83       100
 Michigan St       0.84      0.82      0.83       100
   Minnesota       0.78      0.74      0.76        99
    Nebraska       0.45      0.36      0.40        94
Northwestern       0.70      0.68      0.69        98
     Ohio St       0.53      0.50      0.52       100
     Penn St       0.65      0.74      0.69       104
      Purdue       0.64      0.61      0.63       100
     Rutgers       0.38      0.49      0.43       100
   Wisconsin       0.46      0.29      0.36       100

    accuracy                           0.63      1393
   macro avg       0.62      0.63      0.62      1393
weighted avg       0.63      0.63      0.62      1393
```

The main commonality between these teams is their primary color is red. Most of the other teams in the Big Ten have their own primary color or only share their primary color with one team. Below you can see the accuracy of the predictions by team, where the bar color represents the team’s primary color.



Clearly, having several teams with red confuses the model and makes it more difficult for the model to predict the correct team.

Conclusion

While one would obviously hope the accuracy of the model would be 100%, this model was able to accurately classify images at a rate nine times more than random chance. The model did particularly well for teams with unique colors. This could be used to at least reduce the amount of work needed to classify the images by team.

Assumptions

The integrity of the data was the biggest assumption in this project. This data was not pre-labeled but gathered through Google Images. This means the data was reliant on Google returning acceptable images for each search term. For example, it was assumed a search for "Northwestern Wildcats Football" returned images related to Northwestern Football, not wildcats found in the Northwest. While a quick scan was done, it is difficult to ascertain 7,000 images were accurately classified.

Limitations and Recommendations

The greatest limitation for this project was the ability to collect images. While team websites may have included pictures, there never appeared to be any centralized sets of photos. The only locations photos could be found in mass were Google and stock photo websites like Getty Images. This ultimately led to smaller data sets for each team as Google seemed to only produce around 600 images per search.

A recommendation for future model development would be to obtain more photos to increase the size of the data set. This should allow the model to increase its accuracy, particularly for teams with red as the primary color.

Future Use and Implementation

The most practical applications of this project would be to classify teams at a large scale. Whether this is associating images with current stories or if an organization, such as the Big Ten, received a "photo dump" of older photos. These photos would likely be difficult to sort through.

The implementation of this model is straightforward. An organization could set up a method for receiving images. The model could be set up to process images from this location. The model would then classify all images. Depending on the organization, the model would likely need to be retrained using a wider group of teams.

Ethical Assessment

I do not believe there is any ethical concern in training and using a model to classify football related images. The ethical issues come from obtaining and using images. Many websites consider data scraping to be against the terms of service. This line is further skewed when using a web driver. This isn't the same thing as scraping data but is more of a simulation of someone using a web browser. Another potential ethical issue with using the images is infringing on copyright ownership. However, the use of these images could be considered as educational where copyright laws do not apply. Finally, the issue of Name, Image, and Likeness (NIL) could be a concern. Athletes now have the ability to profit off their NIL and using the images of athletes without their permission could draw criticism.

References

1. Chollet François. (2018). *Deep Learning with Python*. Manning Publications Co.

Questions

1. Why did you choose Big Ten teams as your data set?

   Nebraska is my favorite team so I am most familiar with teams in the Big Ten conference.

2. Was there a team where images were easier to obtain?

   The blue blood teams Nebraska, Ohio State, and Michigan seemed to provide more images. Likely because they have more media covering them than other teams.

3. Was there a team where images were more difficult to obtain?

   For some reason, the web driver had difficulty with Indiana, Michigan State, and Purdue. The web driver would stop around 250 images. I believe it had to do with some sort of time out issue.

4. What kind of images did you use?

   Most of the images are of players during games. Other common images were helmets, stadiums, and logos.

5. Were certain types of images easier to classify? (Individual, Team, Helmet, etc.)

   Unfortunately, it would have required to much time to try and identify different types of images within each team, so this was not assessed.

6. Was there a team that was easiest to classify?

   The model was able to classify Iowa images with the highest accuracy. This is likely due to the uniqueness of yellow as their primary color. Their secondary color is black which is only used by a couple of other teams in the conference.

7. Was there a team that was the most difficult to classify?

   The model struggled with almost all teams with a primary color of red. It is unsurprising the model struggled with Nebraska and Wisconsin the most as their colors and uniforms are very similar.

8. How did you deal with images containing two teams?

   There was no special consideration for images with two teams. The images were classified for each team based on the results of the search terms.

9. Was dropout considered?

Dropout was considered. However, overfitting did not seem to be a big enough issue to justify using a dropout layer.

10. Why did you land on using 60 epochs?

    I did struggle with this decision as it seemed 80-epochs did benefit from more training as it had higher training and validation accuracies. However, the differences did not seem large enough to be worth the potential overfitting.