Kevin McGovern

MSDS 462 Final Project

iOS App for Playing Card Detection on the Edge

Executive Summary

Object detection can be a useful technique to quickly determine the individual objects in a camera frame. Technologies like You Only Look Once (YOLO) [1] have shown users how powerful these models can be to quickly detect distinct objects. Playing cards provide a unique challenge in object detection as many of the cards look very similar across both numbers and suits. Using an iPhone or mobile device to detect which type of card passes in front of a camera could provide several benefits and be used by both professionals and amateurs looking to track cards the cards. The object detection model in this project was trained on a playing card dataset and deployed to an iPhone. The resulting application detects cards with a high degree of accuracy while providing predictions extremely quickly.

Data Creation and Model Approach

One of the first steps to building a machine learning model is data collection. The dataset for this problem had to be created manually due to the lack of online resources. Additionally, in order to make sure the model generalized well to different environments, different lighting conditions, focus levels, and backgrounds need to be used to make sure that the model would work well when deployed. To begin the dataset creation, video files had to be created for each card. Fortunately, there was some existing research in this space that helped guide the creation of images to be used for each class, as they were extracted from the video [2]. Additionally, the previous research included methods to combine the images into two and three card scenes overlaid on top of random backgrounds.

Once the scenes were created, several different modeling approaches were tested. Apple's Create ML provided a good baseline model using the data. However, the accuracy of the model did not exceed 45% after 8 hours of training. In an effort to utilize transfer learning and start of the art object detection, YOLO was attempted next. A model was successfully built using this model as a basis, however, due to the complexity of the architecture converting the model from TensorFlow weights to Create ML proved to be extremely difficult. Finally, another Apple solution was discovered, Turi Create. Turi Create is a Python package that can be run on a Linux machine. This allows for the model to be run on more sophisticated hardware, even taking advantage of GPUs. However, to use this library, further dataset curation had to be done to put this data in a sFrame. That took a bit of API Documentation reading but once that was done, the model was ready for training.

Solution

Before training the model, the data was split into standard train and test datasets (80/20 split). The model was then trained for 10,000 iterations which took just over four hours. When validated, the fitted model provided an average precision of 91% across all of the classes on the test dataset. Appendix 1 displays a chart of the results across all of the playing card numbers and suits. The model was then outputted to Create ML format.

After the model was trained, it was then ready for deployment on an edge device. Using a MacBook Pro and a sample XCode project for object detection (the Breakfast Finder app) [3], the model was deployed to an iPhone XR. The resulting application quickly identifies cards that come into view with a high degree of confidence.

Recommendations for the Future

Given that the outcome is a high performing model, let's conclude with a few recommendations for how to potentially enhance this application even further. One area that the application may struggle with is in the generalization of different playing cards (i.e. cards from another manufacturer). To make sure that the model can detect cards of different styles, other decks could be used in the image and video samples. Additionally, there were some interesting results when the application viewed surfaces without any playing cards on them. When application was pointed at surfaces with colors that are similar to playing cards, the model predicted that it saw playing cards. To resolve this, the model could be retrained using some images with red and black text but a blank label associated label. This would hopefully train the model to generalize to whether something is or isn't a playing card.

Finally, there is an interesting extension that would put this model into practical use. With an application that detects cards, the model could use the information gathered to learn about the context of the scene it is observing. For example, the application could be used to observe a poker table, then given the cards observed it could predict the likelihood of a player winning. The prediction piece of could be accomplished using either supervised learning or reinforcement learning to compute the probability of winning given the cards that are on the board. Overall, the application that was created gives us some great results and provides some interesting possibilities for how to extend this in the future.

Project Links

- Link for Google Colab Folder
  - Initial Dataset Creation
  - Turi Create sFrame Creation
  - Turi Create Model Development
  - Turi Create Model Evaluation
- Link for GitHub Repo
- Link for YouTube video of Demo

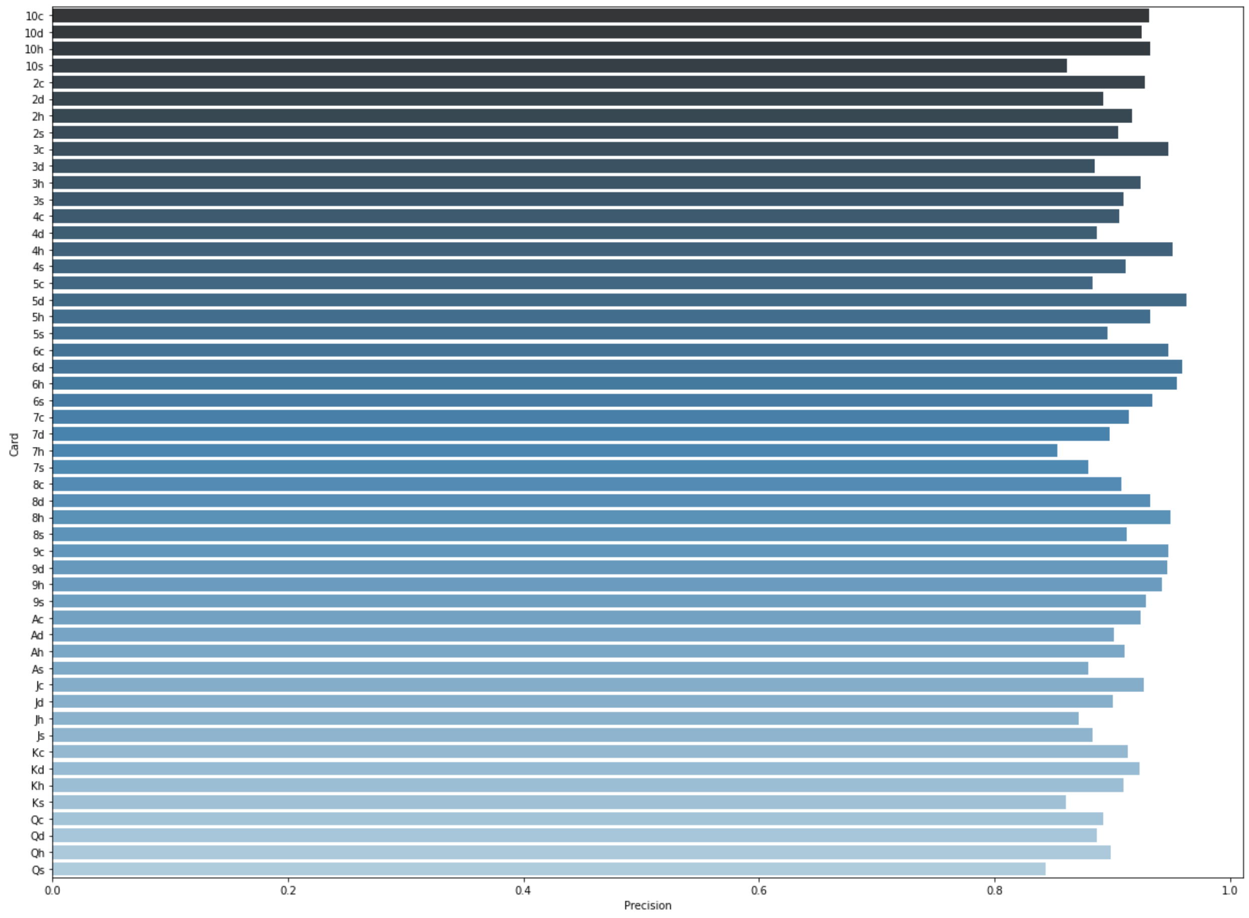
References

[1] https://pjreddie.com/darknet/yolo/

[2] https://github.com/geaxgx/playing-card-detection

[3] https://developer.apple.com/documentation/vision/recognizing_objects_in_live_capture

Appendix 1: Model Evaluation


Card
10c
10d
10h
10s
2c
2d
2h
2s
3c
3d
3h
3s
4c
4d
4h
4s
5c
5d
5h
5s
6c
6d
6h
6s
7c
7d
7h
7s
8c
8d
8h
8s
9c
9d
9h
9s
Ac
Ad
Ah
As
Jc
Jd
Jh
Js
Kc
Kd
Kh
Ks
Qc
Qd
Qh
Qs
Precision