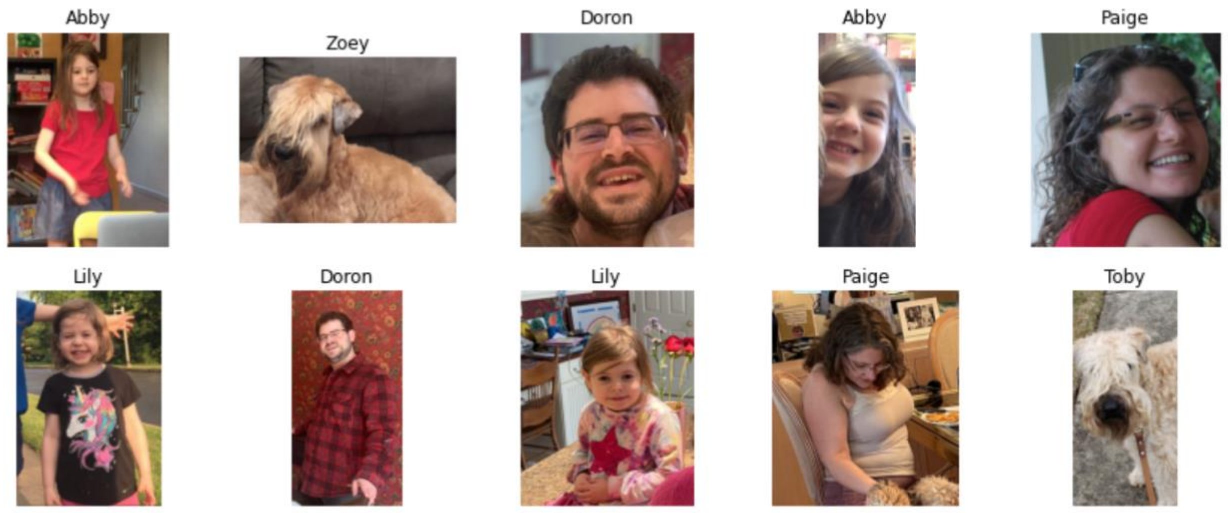
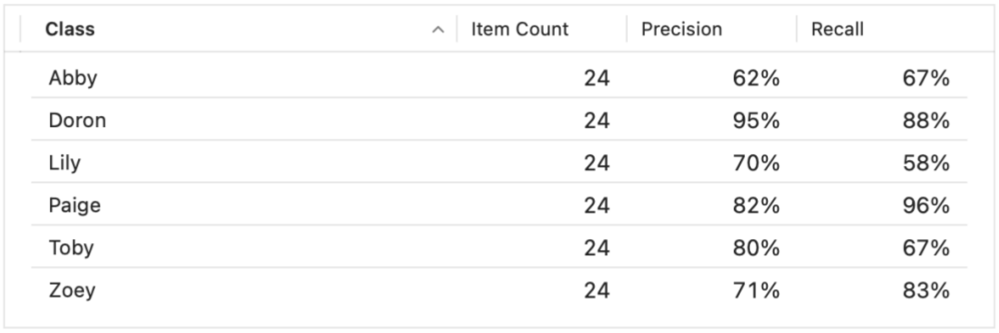
Research Project Synopsis:
Family Member Image Classifier

For this project, I sought to build image classifier models that could take in either live footage or still photos on an iPhone camera and tell me in real time which member of my family was depicted in the image. To do so, I built two models—one trained using automated machine learning (AutoML) and one trained manually using transfer learning—and implemented them in two separate iPhone apps as an Apple CoreML model and a TensorFlow Lite model, respectively.

The dataset for this project was curated from my personal photo library. I searched for photos of my two daughters Abby and Lily taken within the last one year, and from the last two years I found photos of my two dogs Toby and Zoey, my wife Paige, and myself (Doron). After finding 114 photos of each of the six of us, I cropped each photo to only feature only one individual and split the dataset into 90 training images and 24 test images for each class. Some sample images are depicted below:

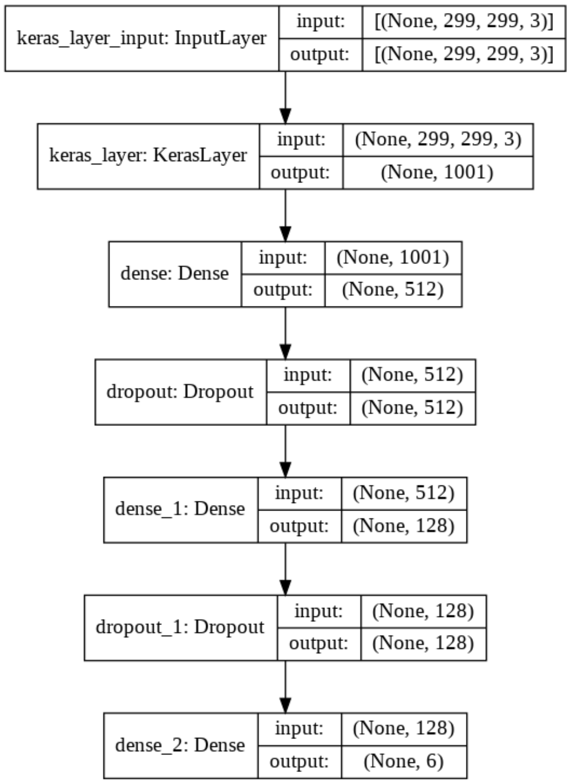


I first trained a model, Model 1, using Apple's AutoML tool CreateML, and implemented it entirely within the Apple universe by creating a Core ML model object, building an application in Xcode using the Swift programming language, and deploying it to my iPhone 12. This Model 1 achieved 76% accuracy on the test set, broken down as follows:

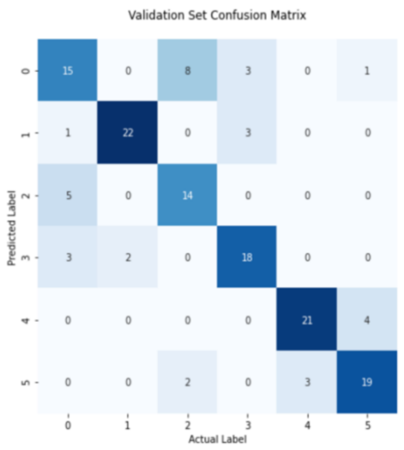


Class	Item Count	Precision	Recall
Abby	24	62%	67%
Doron	24	95%	88%
Lily	24	70%	58%
Paige	24	82%	96%
Toby	24	80%	67%
Zoey	24	71%	83%

For my second model, Model 2, I trained a model manually using TensorFlow and Keras on the Google Colab platform. To do so, I used transfer learning, extracting features from the input images using the InceptionV3 model and layering on top a handful of dense and dropout layers to predict my six classes, as depicted to the right. Finally, I optimized the architecture, dropout rate, and batch size. I converted this model to a TensorFlow Lite (TF Lite) model object, built a second application using Xcode and Swift, and deployed this one as well to my iPhone. This model also achieved a 76% accuracy rate on the test set, matching the CreateML model, as seen in the below classification report. Interestingly, both models performed similarly, with similar strengths and weaknesses.

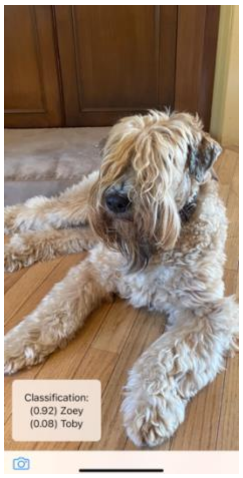


```
Validation Accuracy: 0.7569

Validation Classification Report:
              precision    recall  f1-score   support

           0     0.6250    0.5556    0.5882        27
           1     0.9167    0.8462    0.8800        26
           2     0.5833    0.7368    0.6512        19
           3     0.7500    0.7826    0.7660        23
           4     0.8750    0.8400    0.8571        25
           5     0.7917    0.7917    0.7917        24

    accuracy                         0.7569       144
   macro avg     0.7569    0.7588    0.7557       144
weighted avg     0.7633    0.7569    0.7582       144
```

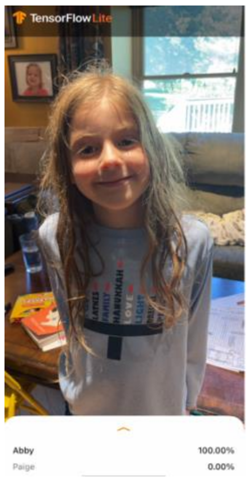


The iPhone application that implements Model 1—the AutoML model—presents users with the choice of either selecting a photo from their photo library or taking one live, and after a photo is either selected or taken, the app makes a prediction as to which of my six family members is depicted. The display shows the top two most likely predictions along with their associated probabilities, as seen in the screenshot from the app below (left). On the other hand, the iPhone application that implements Model 2—the TensorFlow Lite model—does not require a still photo but instead takes in live footage using the camera and updates its on-screen predictions and associated probabilities in real time, as seen below (right):

<u>Model 1 App</u>



<u>Model 2 App</u>



Of course, there are several next steps to take this project—both the model and the implementation—to the next level of functionality. On the modeling side, data, more than anything else, is what will improve the model the most. Right now the model is confusing the two dogs about one out of every six times, but the most common error it is making is mixing up my two daughters. That is an understandable enough mistake, as similar as they look at this young age, but continuing to curate the dataset, gathering images of each of my six family members in multiple and different settings, is what the model really needs to achieve a higher accuracy. On the implementation side, my next step is to improve the user interface of my two apps. I attained a basic level of comfort and familiarity with Xcode and Swift while doing this project, but I would like to invest additional time improving my Swift skills to improve the look and feel of the applications.

As the world moves towards using more automated machine learning and pre-trained models, this project provided very practical experience with both AutoML and transfer learning. But even moreso, the experience with productionalizing the models and deploying them as iPhone applications was invaluable.