SafeCabinet

Automated Locking Storage System

Devin Quinn
Electrical and Computer
Engineering
SUNY Binghamton
dquinn6@binghamton.edu

ABSTRACT

In recent years, the field of deep learning has exploded and it is quickly changing today's technology. As a tool for solving various types of problems, deep learning influences many different disciplines and fields. Some example problems deep learning models can solve are image classification, facial recognition, natural language processing, and recommendation systems, to name a few. Deep learning is being applied to homes to create "smart" appliances for everyday items. For example, a smart oven may use deep learning to predict what kind of food you are cooking, and automatically set a cooking time based on the predicted food.

1 INTRODUCTION

Safety is the number one priority when having children present in the house. It is good practice to keep dangerous tools and substances out of children's reach by storing them in locked cabinets or hard to reach places. However, this provides an inconvenience, and may not even be possible in some households. This is the motivation behind SafeCabinet, an automated locking storage system for households designed to be used in kitchens, bathrooms, or any other household room where dangerous items need to be secured.

SafeCabinet's camera tracks who is trying to access the cabinet and uses deep learning to predict the age of the individual. If it predicts the individual to be a child, or if the room is empty, the cabinet remains locked. If the prediction is an adult, the cabinet unlocks.

SafeCabinet is designed to be lightweight for embedded system integration and hands-free for user convenience. Deep learning inference occurs on the device for quick response times.

2 DESIGN

The system consists of three main components: a camera used for sensing, a central processing board that performs all of the system's data computations and device communication, and a solenoid lock used for locking the cabinet. A data-flow diagram of the system is shown in Figure 1.

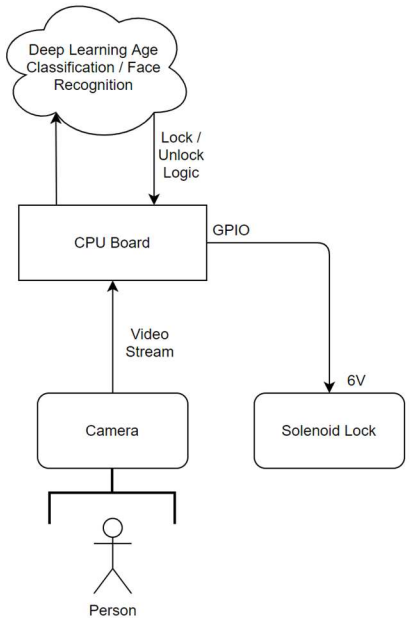

Figure 1: System Data-flow Diagram

2.1 HARDWARE

The system's main component is TI's AM574x Industrial Development Kit (IDK) [1]. This development board can be used for a wide range of

applications in industrial control and communication, automation, robotics, etc. The board not only serves as the central processing unit for all of the system's peripheral devices, but also hosts the deep learning model. It contains 2GB of DDR3L SDRAM, a 70MB microSD card, 16GB of eMMC NAND Flash and 32MB of SPI NOR Flash. In addition to the AM5748 processor that contains dual ARM Cortex-A15 and C66X DSP cores, the board also has two embedded vision engines (EVEs) that provide acceleration of deep learning algorithms. Since the deep learning inference occurs on-device, the large amount of memory, powerful processing cores and additional EVEs make the AM574x IDK an ideal development board for this project. A top view of the AM574x IDK is shown in Figure 2.

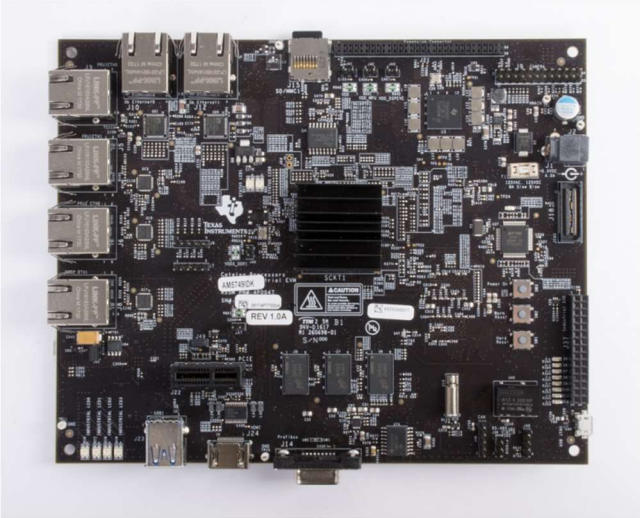

Figure 2: TI AM574x Industrial Development Kit

The OmniVision OV2659 SoC camera [2] is used as the system's sensing device and connects directly to the development board. It supports 720p HD video recording at 30 fps and low-light sensitivity, making it a good sensor for general use in any room.

Lastly, a 6V electromagnetic solenoid lock [3] is used as the system's locking mechanism. The lock connects to the board through one of the GPIO pins. When the line is low, the circuit is off and remains unlocked. When the line goes high, the circuit is on and locks. The lock is small and lightweight, making it easy to be mounted on a cabinet door.

2.2 SOFTWARE

The system runs a Linux ARM OS on a microSD card. The entire OS filesystem can be accessed by mounting the microSD card on a Linux virtual machine, and all programming for the project is done in the virtual machine and put directly onto the card. When the SD card is on the board and the system is live, the OS can be accessed through its Matrix GUI by connecting an HDMI cable from the board to a display monitor.

To perform age classification, a TensorFlow deep learning model is developed using a Google Colab Notebook connected to a local runtime. Transfer learning techniques are used during development; an open-source model pretrained on ImageNet serves as the base to increase optimization convergence and decrease training time. MobileNetV1 [4] is chosen as the base model as it is designed to be lightweight and computationally efficient, making it ideal for embedded systems. The first 30% of the model's inner layers are frozen to preserve initial low-level weights, while the rest of the model is trainable. The head of the model includes a 2D convolution layer, a 2D global average pooling layer and a dropout layer for regularization, and a dense layer for prediction outputs. For each input image, the model outputs a two-element array where the value at index 0 corresponds to the predicted probability of the image belonging to class 0 (LOCK) and index 1 corresponds to the predicted probability it belongs to class 1 (UNLOCK). A summary of the model is shown in Figure 3.

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
mobilenet_1.00_224 (Function (None, 7, 7, 1024)        3228864
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 5, 5, 32)          294944
_________________________________________________________________
global_average_pooling2d_2 ( (None, 32)                0
_________________________________________________________________
dropout_8 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 66
=================================================================
Total params: 3,523,874
Trainable params: 3,469,922
Non-trainable params: 53,952
_________________________________________________________________
```

Figure 3: Deep Learning Model Summary

3 IMPLEMENTATION

For model training, a dataset with 28,958 total images consisting of two open-source labeled datasets is used, the UTKFace dataset [5] and the House Rooms Image Dataset [6]. The images are sorted into two classes: LOCK and UNLOCK. The age threshold is set at 11; images of individuals under the age threshold and images of house rooms are classified as LOCK while images of individuals over the age threshold are classified as UNLOCK. With this labeling, the trained system will unlock for individuals over 11 and lock for children under 11 and empty rooms. After sorting the images, there are 20,425 images of older individuals, 3,283 images of younger individuals, and 5,250 images of rooms. Since this is an imbalanced dataset biased towards older individuals, image augmentation techniques are used via the Albumentations library [7]. In the image augmentation pipeline, images are replicated and minor random adjustments such as shifts, rotates, and crops are performed to augment the dataset with more images. The images are also resized to 224x224 and split into three sets: training, evaluation, and test.

The deep learning model is compiled with categorical cross entropy loss and the Adam optimizer. After training on the augmented training and evaluation sets for 20 epochs, the model is evaluated on the test dataset and results in 97.57% accuracy. The prediction results on a batch from the test set are shown in Figure 4.

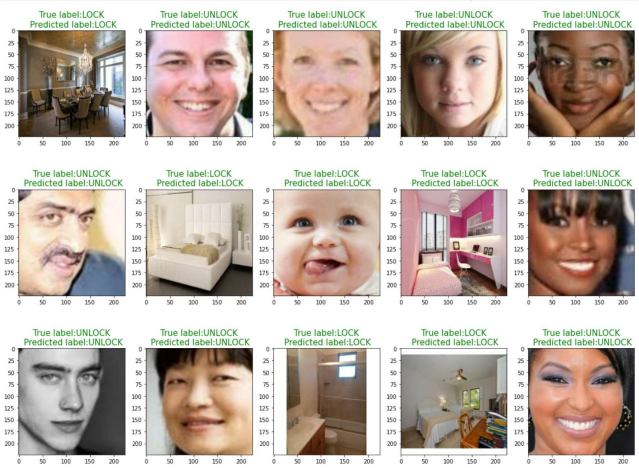

Figure 4: Model Prediction Test Results

The model is now ready for on-device inference, but first it is converted to a TensorFlow Lite model [8]. The TensorFlow Lite format optimizes for on-device machine learning in terms of size, power consumption, and performance, making it ideal for embedded systems. After conversion, the model is put onto the SD card.

To run the model on the system, the *tflite_classification* executable [9] is used. It takes a TensorFlow Lite model, label list, and image file/camera video stream as inputs, and runs the model on the device.

Unfortunately, the GPIO pins could not be accessed and therefore the solenoid lock could not be used during implementation. Therefore, the system was modified to have the output driven through the HDMI and displayed on a monitor rather than through a GPIO line. Several scripts were developed to have this model run as a camera application on the Matrix GUI.

4 RESULTS

When presented with images from each of the three classes (adult, child, room), the model is able to correctly predict whether the system should lock or unlock. Figures 5, 6, and 7 show the results of the model with the camera fixed on a transitioning slide show. First, a child's face is shown, and the model correctly outputs LOCK with high probability. Then, the model is shown an adult's face, and the output changes to UNLOCK. This output has lower probability, but it has still made the correct prediction. Finally, an image of a kitchen is shown, and the model correctly outputs LOCK again.

Figure 5: Model Prediction for Child Image

Figure 6: Model Prediction for Adult Image

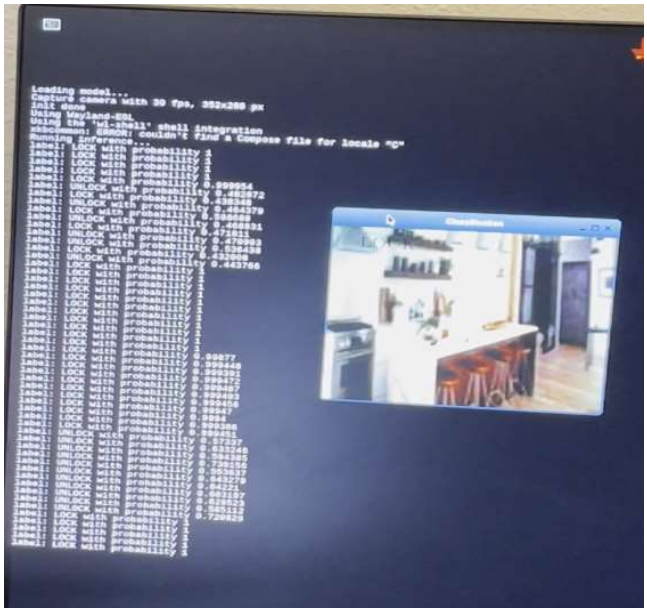

Figure 7: Model Prediction for Room Image

5 CONCLUSION

Although the lock had to be removed from the final implementation, the results were promising and show that the system works correctly when presented with various images in real-time. If the outputs of this system were to be connected to a solenoid lock as originally intended, the cabinet would successfully lock and unlock based on the camera input. The next steps are to apply this design to a physical storage system, making SmartCabinet another smart device for your home.

REFERENCES

[1] AM574x Industrial Development Kit
https://www.ti.com/tool/TMDSIDK574

[2] OmniVision OV2659 Camera Sensor
https://www.digchip.com/datasheets/3262771-ov2659-color-cmos-uxga-2-megapixel.html

[3] Uxcell DC 6V 1.5A Electromagnetic Solenoid Lock
https://www.amazon.com/uxcell-Electromagnetic-Solenoid-Assembly-Electirc/dp/B07TKR7KLH/ref=sr_1_8?dchild=1&keywords=solenoid+lock&qid=1620654711&sr=8-8

[4] MobileNet
https://github.com/tensorflow/models/blob/master/research/slim/nets/mobilenet_v1.md

[5] UTKFace Dataset
https://www.kaggle.com/jangedoo/utkface-new

[6] House Rooms Image Dataset
https://www.kaggle.com/robinreni/house-rooms-image-dataset

[7] Albumentations
https://albumentations.ai/

[8] TensorFlow Lite
https://www.tensorflow.org/lite

[9] TI TensorFlow Lite Demo
https://git.ti.com/cgit/apps/tensorflow-lite-examples/tree/