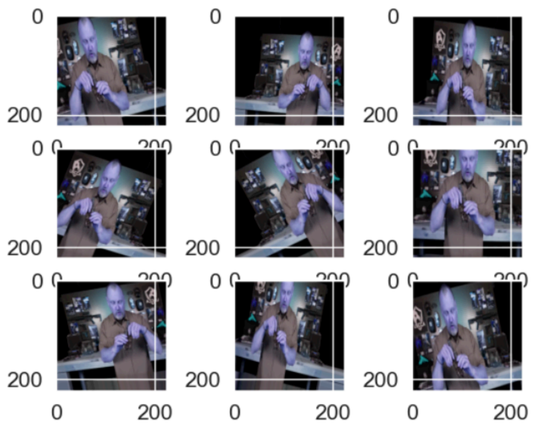
Image Classification on Weapon Detection

Li Jiayao (A0160257J)
National University of Singapore

Introduction

Image classification has vast applications in the real world, one of which is weapon detection (which includes handgun, and knives). Weapon detection helps to enhance the current surveillance and control system. The purpose of this project is to train a model that can classify images with weapons and without a weapon.



Methodology

1. Baseline Model

The baseline model is built as a simple neural network to solve the given classification task. The architecture of the neural net is shown as follows:

The image is first reshaped into (128, 128, 3). I simply use three dense layers in the network, and employ sigmoid activation function at the last layer to output a single probability. This probability output can be interpreted as how likely this image contains weapons. If the probability output is more than 0.5, I classify the image as label of weap, else norm. Overall the model achieved an accuracy of 85% on training and validation dataset and 60% accuracy on test. For the following section, I seek for improvement on the accuracy of the test data.

Methodology

2. Improved Model

Shifting away from this fully connected layers of network, I want to improve the model with convolutional layers. The main advantage is that the spatial structure is preserved. Instead of stretching the image array into one vector to be multiplied with the weights as in our baseline model above, the 3-dimensional structure is kept in the new convolutional neural network.

For this CNN model, we are inspired by the VGG architecture, and added following improvements:

a) Data augmentation is used to enlarge our training dataset based on the existing dataset by randomly rotating, shifting, zooming, and flipping.

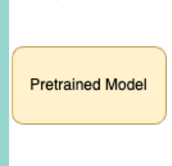
b) Batch Normalisation helps to prevent overfilling, allows us to use higher learning rate and reduce training time.

c) 3*3 Kernel: stack 2 layers of convolutional layers before pooling creating an effective receptive field of 5*5. Deeper network helps to bring better results.

e) Learning rate and epochs: to train the model for better performance, we starts with small learning rate with decay, and longer epochs with early stopping to avoid overfitting.

3. Pre-trained Model

VGG16 and MobileNetV2 are experimented for this project as well, and both achieved ~70% accuracy. I first freeze the model and train only on the last two layer of 1024 fully connected nodes and one output layer with sigmoid activation function for 5 epochs. Then reduce the weights by 0.01, and unfreeze the model to train for another 10 epochs.

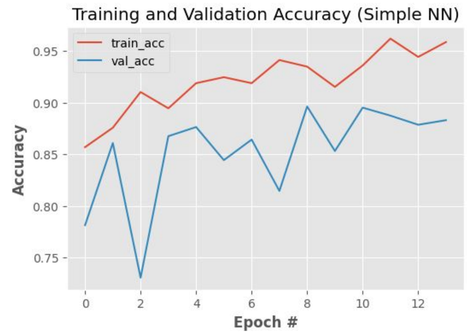


	precision	recall	f1-score	support
0	0.77	0.75	0.76	1271
1	0.61	0.63	0.62	775
accuracy			0.70	2046
macro avg	0.69	0.69	0.69	2046
weighted avg	0.71	0.70	0.70	2046

Results

1. Baseline Model

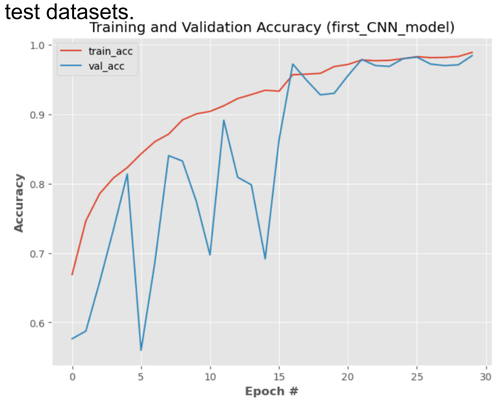
The following is the visualisation of accuracy in train, validation and test dataset.



	precision	recall	f1-score	support
0	0.65	0.78	0.71	1271
1	0.46	0.31	0.37	775
accuracy			0.60	2046
macro avg	0.56	0.54	0.54	2046
weighted avg	0.58	0.60	0.58	2046

2. CNN Improved Model

The results shown below is the accuracy in train, validation and test datasets.

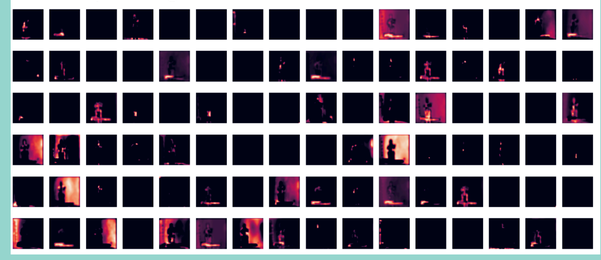


	precision	recall	f1-score	support
0	0.79	0.77	0.78	1271
1	0.64	0.66	0.65	775
accuracy			0.73	2046
macro avg	0.71	0.72	0.71	2046
weighted avg	0.73	0.73	0.73	2046

Results

3. Activation Map Analysis

Displayed below is some activation maps at the last convolution layer in our self-built CNN model. We can see that the activation maps capture the contour of how people grab the handgun which can be used to infer whether the image contains a weapon. Also, some activation maps are totally black which means that the pattern encoded by the filters were not found in the input image. Most probably, these patterns must be complex shapes that are not present in this input image



Conclusion

In conclusion, in this project, I attempted to train 3 different types of models. Starting from a very basic fully connected models to a convolutional neural network, I observed 13% accuracy increment on test dataset. Lastly, I trained the pretrained models to bench mark my result. In the pretrained models, ~70% accuracy on dataset is also observed.

Acknowledgements

1. https://developers.google.com/machine-learning/practica/image-classification
2. http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture5.pdf
3. https://arxiv.org/abs/1502.03167
4. https://github.com/keras-team/keras-applications/blob/master/keras_applications/vgg16.py
5. https://towardsdatascience.com/visual-interpretability-for-convolutional-neural-networks-2453856210ce
6. https://www.youtube.com/watch?v=DAOcjicFr1Y&t=1656s