Kevin Ferreira
CAP 6619

# I. Introduction

Deep learning is a branch of machine learning that uses neural networks to find complex patterns in large datasets. For example, deep learning has enabled us to identify visual patterns directly from data rather than creating features, allowing us to use image classification which is essential in computer vision (1).

One example of deep learning is the convolutional neural network (CNN), a model designed to process grid-like data such as images. CNNs are effective for image classification because they use filters to capture patterns while reducing the number of parameters. (2) Another feature that CNNs can have is pooling layers which reduce spatial dimensions while retaining information (3).

For this study, a CNN will be used to distinguish the images of cats and dogs. This problem is one of the most widely used and is quite challenging because both dogs and cats share similar traits, which in turn requires robust models to distinguish them accurately (4). For problems like this, CNNs also use dropout layers, which serve as a regularization technique by randomly disabling parts of neurons to prevent overfitting. (5) The motivation of this study is to understand how the architecture of a CNN can influence the effectiveness of image classification. The goal of this study is to compare two CNNs that will be used for the cat-dog image classification where one will have three convolution layers, two pooling layers, and two dense layers, and the other CNN will use the mentioned layers, but modified to have batch normalization, larger kernels, and dropout layers.

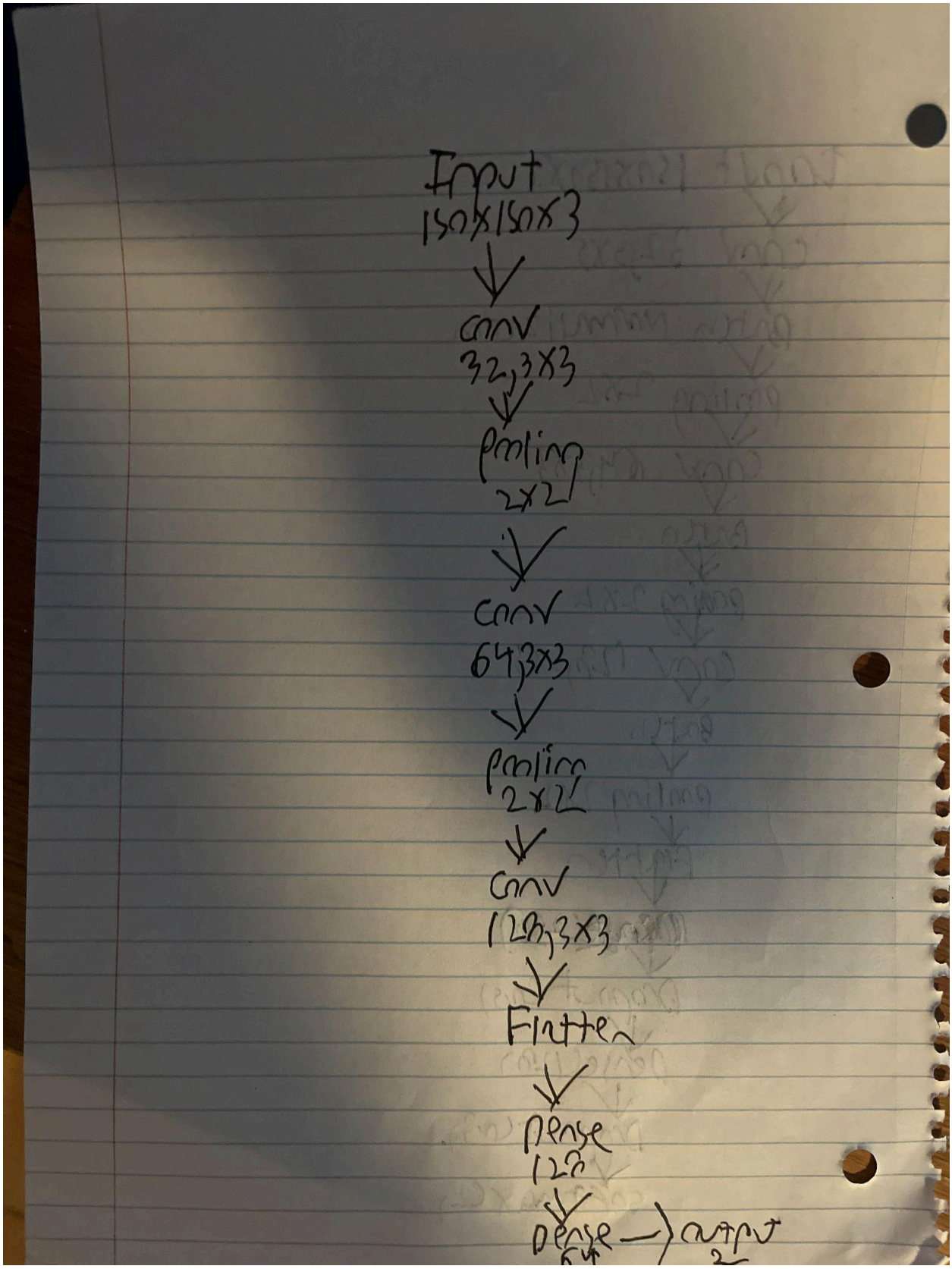
# II. Main Body

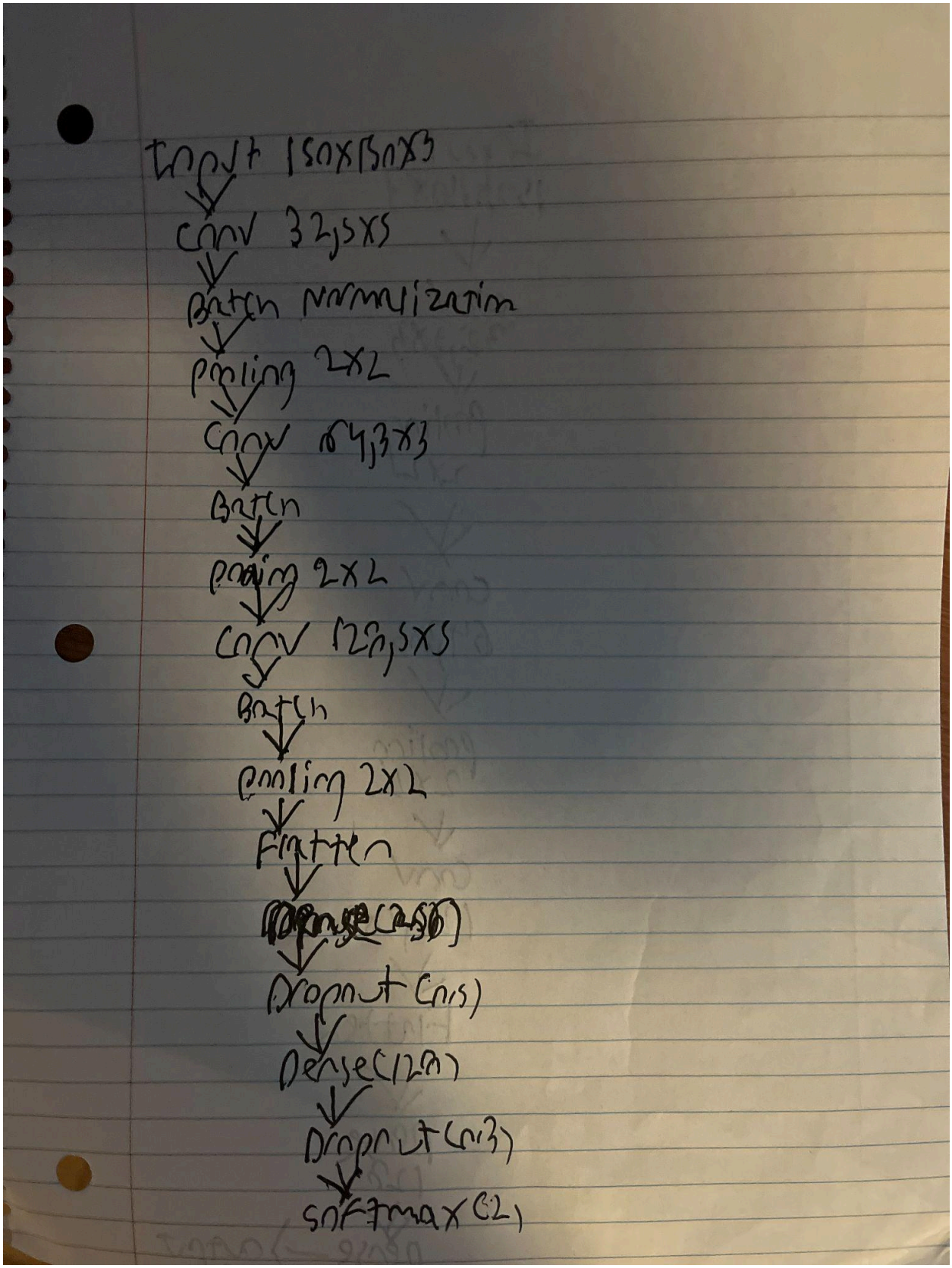
The purpose of this study is to design two CNN models for classifying images of cats and dogs. CNNs operate by applying filters to input images to extract local features, such as patterns. The filters will create feature maps that are processed throughout the models. Having two different CNN models allows us to examine how the

CNN architecture affects image classification performance, including the impact of kernel sizes, batch normalization, and dropout layers.

The first CNN model consists of three convolutional layers with 32, 64, and 128 filters and they use 3x3 kernels and a ReLU activation function. Convolutional layers are the most crucial part of CNNs, as they extract features from input data to form feature maps. Each convolutional layer is followed by a pooling layer which reduces the spatial dimensions without removing information. After passing through the convolutional layers, the data is passed through the flatten layer, which converts multidimensional vectors into a single vector. Then, the data is passed through two dense layers with 128 and 64 units, respectively, to prepare it for classification. This is considered to be the standard version of a CNN.

The second CNN will inherit the same layers as the first, but with some modifications. The first modification was that the kernel sizes of the first and third convolutional layers were increased to 5x5 which in turn increased the receptive field and captured broader spatial contexts. The second modification was to add batch normalization layers after each convolutional layer. The purpose of batch normalization layers is to increase the stability of the learning process and address the covariate shift. Finally, the third modification was to add dropout layers after each dense layer, with rates of 0.5 and 0.3, respectively. Dropout layers help prevent overfitting by randomly dropping a subset of neurons, thereby reducing co-adaptation among neurons. The advantage of this mode is regularization, which helps reduce overfitting. Figure 1 shows the first basic CNN model and Figure 2 shows the second modified CNN model with the modifications.


Input
150x150x3
conv
32,3x3
pooling
2x2
conv
64,3x3
pooling
2x2
conv
128,3x3
Flatten
dense
128
dense
64
output

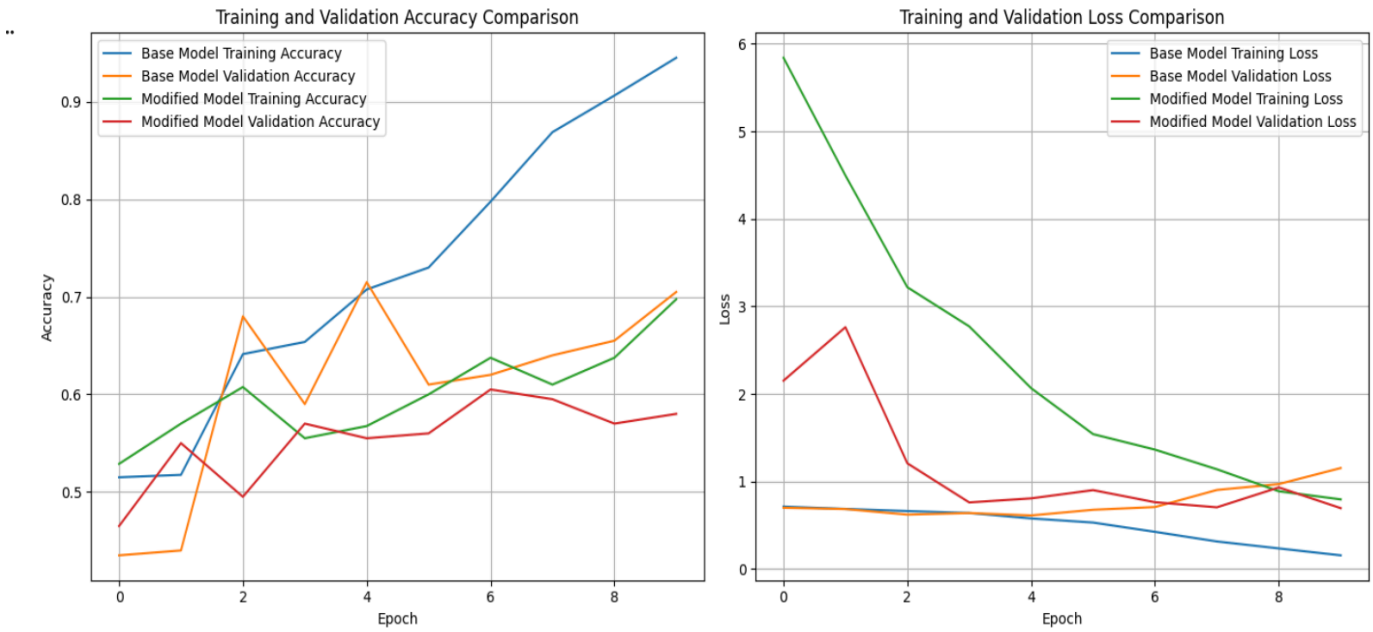

Input 150x150x3
Conv 32,5x5
Batch Normalization
Pooling 2x2
Conv 64,3x3
Batch
Pooling 2x2
Conv 128,5x5
Batch
Pooling 2x2
Flatten
Dropout (0.5)
Dense(128)
Dropout (0.3)
Softmax (2)

# III. Experiments

The entirety of the experiments was written in Python, using the TensorFlow and Keras libraries. The dataset contains 1000 RGB images that are evenly split between dogs and cats. After the images were extracted and loaded, they were resized to 150x15 and the pixel values were normalized to [0,1]. Then, they were split into 80/20 training-validation sets and batch sizes of 32. The classes “cat” and “dog” were also created.

The first CNN model is defined with three convolutional layers, each with a varying number of filters (32, 64, and 128) and 3x3 kernels. A pooling layer followed each convolutional layer, and after the convolutional layer series comes a flattening layer, then two dense layers with 128 and 64 units and ReLU activations. Finally, the output layer uses a softmax for image classification into dogs and cats. The second CNN model uses the same architecture, but with some modifications: the first and third convolutional layers had their kernel sizes increased to 5x5, batch normalization was added after each convolutional layer and before the pooling layer, and a dropout layer was added after each dense layer with rates of 0.5 and 0.3 and the dense layers were also increased to 256 and 128 units. Both models were trained for 10 epochs, and their performance was evaluated using validation loss and accuracy. The following table displays the validation loss and accuracy scores, and the plots of these scores across epochs for both models.

| Score/Model | Basic | Modified |
|---|---|---|
| Validation Accuracy | 0.7050 | 0.5800 |
| Validation Loss | 1.1541 | 0.6957 |




Based on the results, the basic CNN achieved the higher validation accuracy, while the modified CNN achieved the lower validation loss. The basic model correctly identifies the images, but it overfits by memorizing the training data. The modified model has a much lower loss score due to regularization, which combats overfitting, but it sacrifices accuracy in the process.

# IV. Conclusions

Overall, the project explored the problem of classifying images of cats and dogs using convolutional neural networks. Two CNNs were developed for this experiment: the first was a basic model with convolutional, pooling, flatten layers, and dense layers, and the other used the same design but with larger kernel sizes, batch normalization, and dropout layers. The results showed that the basic CNN achieved higher validation accuracy, while the modified CNN achieved lower validation loss. These findings revealed that regularization can reduce overfitting, but it comes at the cost of accuracy. In the future, this study can be improved by expanding the dataset or experimenting with CNN parameter tuning.

# References

[1] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, pp. 436–444, 2015.

[2] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[3] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a convolutional neural network,” in *Proc. ICCC*, 2017.

[4] A. Krizhevsky, I. Sutskever, and G. H. Hinton, “ImageNet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[5] N. Srivastava et al., “Dropout: A simple way to prevent neural networks from overfitting,” *JMLR*, vol. 15, pp. 1929–1958, 2014.