ا

**جامعة قناة السويس**

**كلية الهندسة**

**Neural Networks**

**Project**

**:Presented by**

**احمد حسام جمال الدين**

**زياد محمد جمال**

**مريم سمير السيد وهبه**

**احمد حسانين محمد**

**احمد هاني سعيد يوسف**

**📌 Part I: Data Preparation**

**1. Dataset Source**

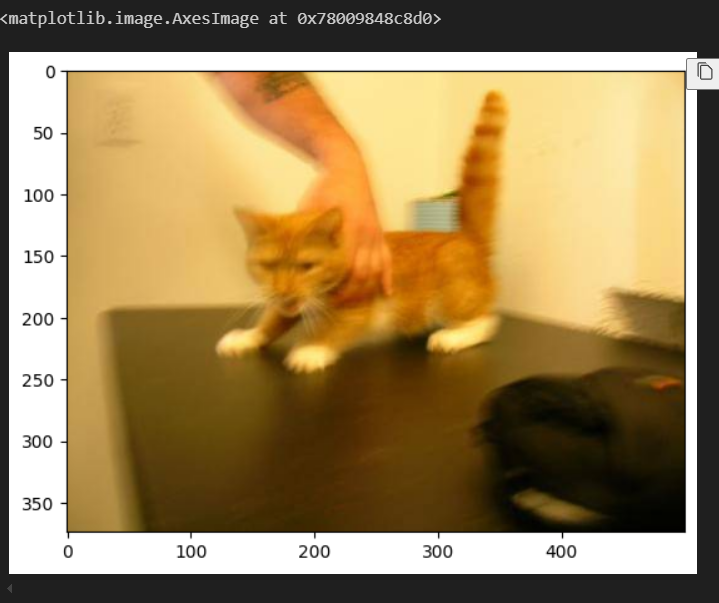
* A dataset of images consists of cats & dogs
* Link to dataset: <https://www.kaggle.com/datasets/sreetejadusi/cats-vs-dogs>

**2. Data Loading**

* Copied the dataset destination to enter it
* Converted the image to a NumPy array and displayed the shape of it

**3. Data Description**

* 3000 Image
* 2 Classes, 1 for cats and the other dogs
* Different image sizes

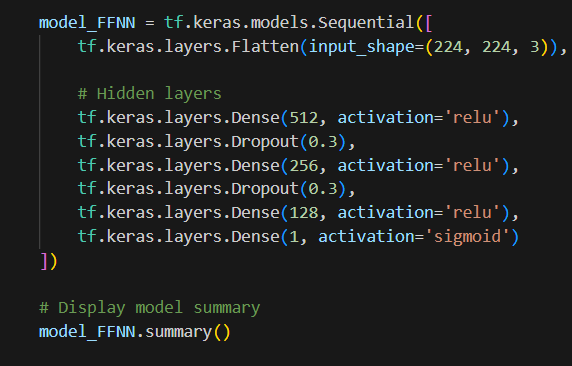
**4. Data Visualization**

* A sample of the dataset:

**🧠 Part II: Training the Models**

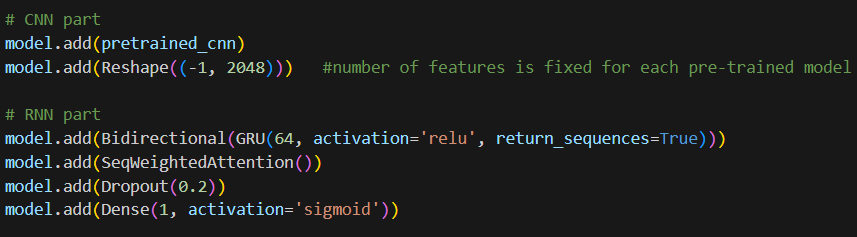
**A. Feedforward Neural Network (FFNN)**

* Architecture consists of 4 layers
* Activation function for hidden layers (Relu) & Output layer (Sigmoid)



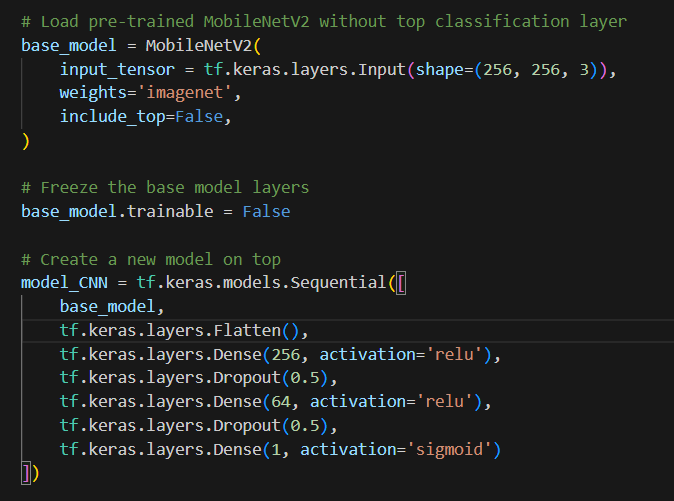
**B. Recurrent Neural Network (GRU)**

* Architecture consists of 4 layers
* Input reshaping for GRU (-1, 2048)
* 3 Hidden layers and 21,673,514 parameters



**C. Convolutional Neural Network (CNN)**

* We used MobileNetV2



**🔧 Hyperparameter Tuning**

**1. Batch Size**

* Batch size 32

**3. Dropout Rates**

* Dropout settings (%50)

**4. Optimizers**

* Optimizers tested (Adam)

**5. Learning Rate**

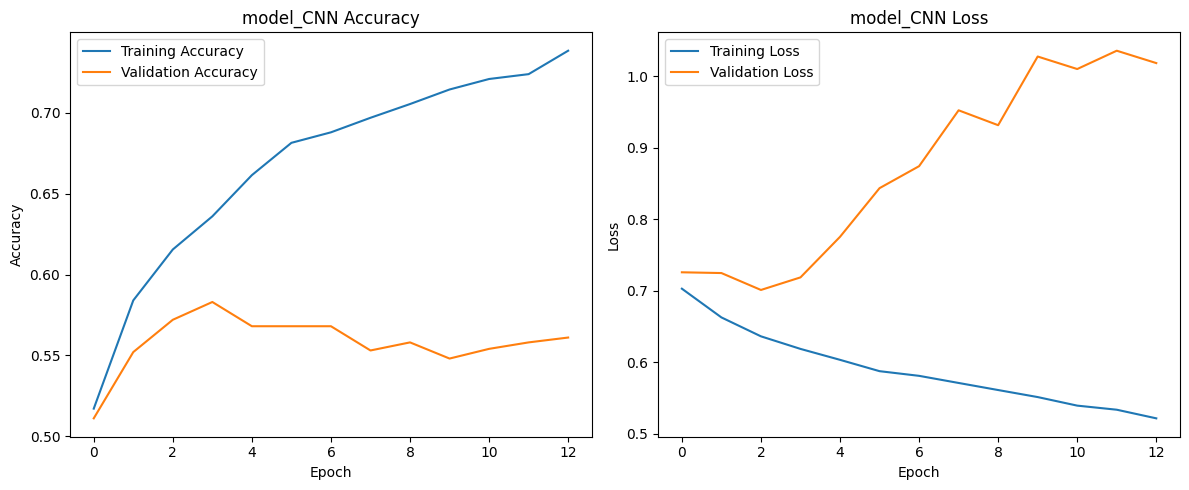
* Lr = 0.00001

**6. Activation Functions**

* We did a comparison between ReLU and Tanh and ReLU was better.

**8. Overfitting/Underfitting Analysis**

* Underfitting



* We adjusted the number of convolutional & pooling layers

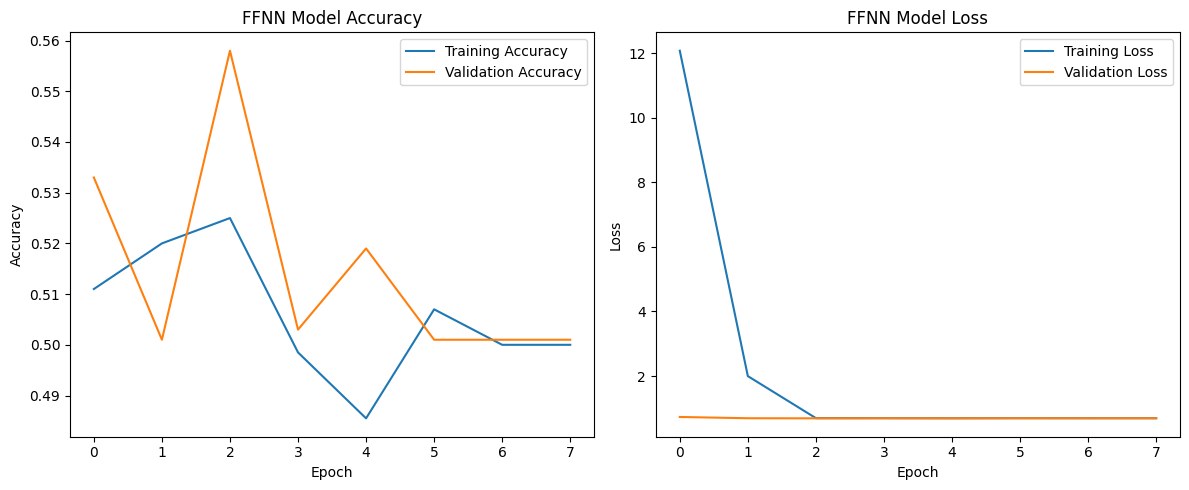
**✅ Part III: Model Evaluation**

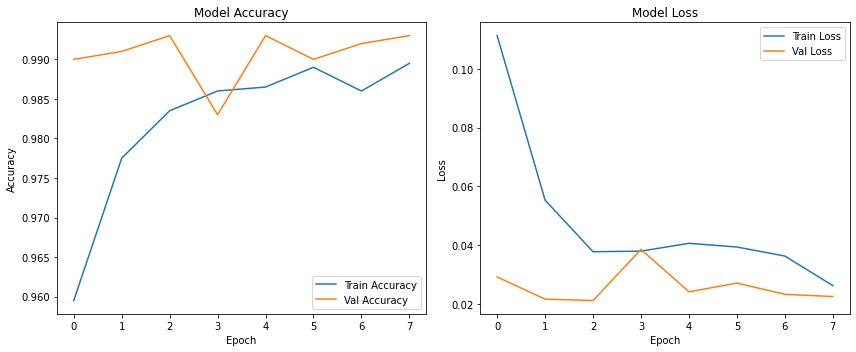
**1. Evaluation Functions**

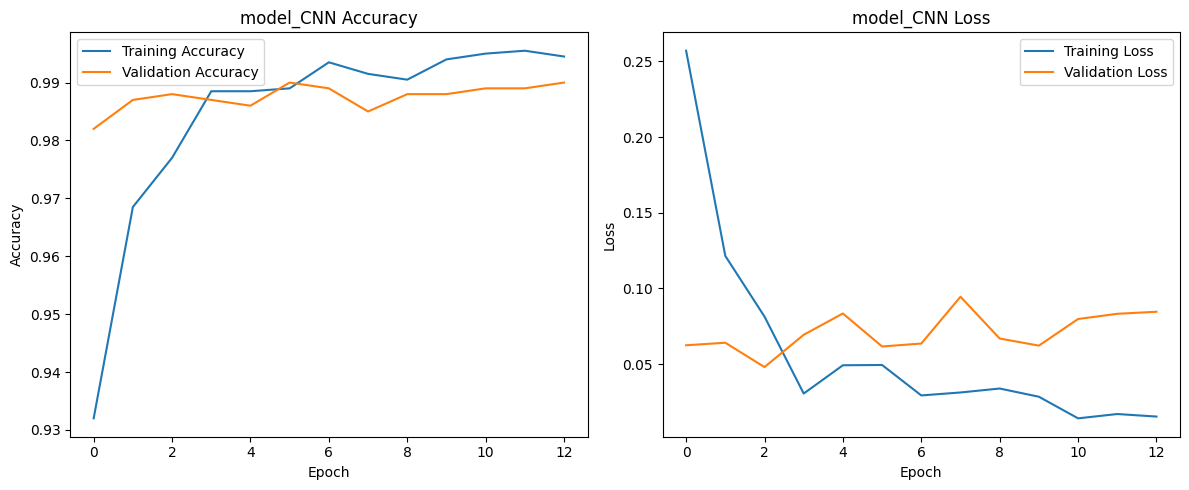
* For each model model performance was evaluated using accuracy as the main metric, and binary cross-entropy as the loss function

**2. Performance Summary**

* FFNN model testing accuracy: 55%, training accuracy: 52%
* RNN (GRU) testing accuracy : 99,3%, training accuracy: 99%
* CNN model testing accuracy: 99%, training accuracy: 99,6%
* accuracy/loss curves for each model respectively:

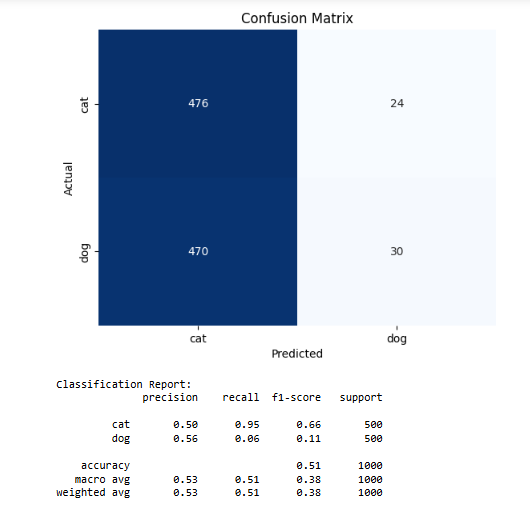


****

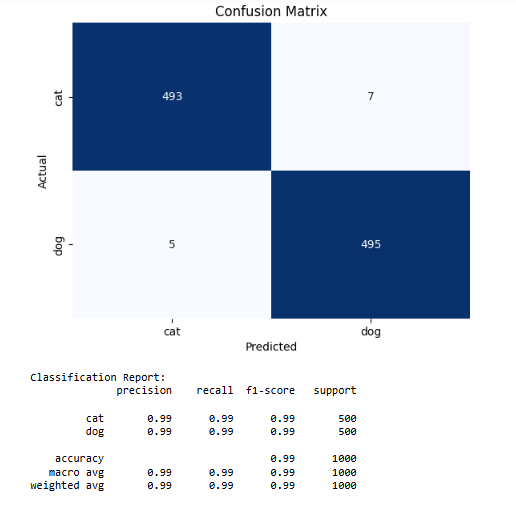
****

**3. Confusion Matrix & Classification Report:**

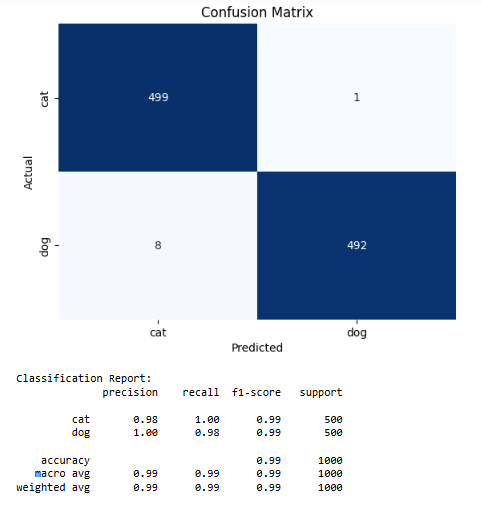
**FFNN:**



**CNN:**



**RNN:**



**📈 Analysis and Discussion**

* RNN model was the best model that gave the best performance.
* RNN works only on sequential data, so how did we overcome this problem to have it work with images?.
* We started by searching the web and Ai on how to overcome this problem.
* So initially we were trying to find a way to make a hybrid RNN & CNN so that it could work with. images and it gave us poor accuracy so we had to find an alternative way to get a better accuracy and a better model.
* Then we decided to try pretrained models, at which we tested 3 models (MobileNetV2, ResNet101V2 ,xception).
* We tried different RNN models (simpleRNN , LSTM , GRU) where GRU came out to be the best model
* Also tried to implement it with and without bidirectional.
* Tested with the activation functions ReLU, Tanh .
* Tried to use SeqSelfAttention and with SeqWeightedAttention.
* Different batchsize , Target size of the image.
* With and without Dropout.
* With and without Augmentation and a very slight Augmentation.