**Data Acquisition: Deep Learning**

**FINAL PROJECT REPORT**

(Gundeep Singh Saluja)

**Problem Statement:**

The purpose of this project is to classify a set of input images as containing dog or cat using Deep Learning libraries like Tensorflow and Keras. Since this is an image classification project, I am going to use Convolutional Neural Network model.

**Dataset Used:**

I am going to use the ASIRRA (Animal Species Image Recognition for Restricting Access) dataset on Kaggle to train my algorithm that contains 25000 images of dogs and cats.

Link: <https://www.kaggle.com/c/dogs-vs-cats>

**Algorithmic Steps Followed:**

1. Loading the dataset:

Split the dataset into training and validation sets using the sklearn library function in the model\_selection class train\_test\_split(). For transforming the images I have used the ImageDataGenerator() function to generate batches of images to fit the data using fit\_generator() function.

1. Mapping the categorical variables:

Used the basic OS module listdir() function to retrieve all the images from the training set in a list and mapping ‘1’ to dog and ‘0’ to cat.

1. Model Building:

Used 3 convolution layers with filters 32, 64, and 128 and activation function RELU. For each layer, I have used Max Pooling of pool size (2,2) to reduce the number of pixels in the output from the previous layer. Also, I have added Batch Normalization to prevent the problem of vanishing gradient after every layer, and in the end, I have implemented regularization through Dropout with a probability of 0.25 in the convolutional layers and 0.5 in dense layers.

1. Training the model:

Used the EarlyStopping() function from Keras callbacks module to terminate the training when the validation accuracy doesn’t seem to increase after 10 epochs. Also used the ReduceLROnPlateau() function to reduce the learning rate by a factor of 0.5 if the validation accuracy doesn’t increase after every 2 epochs. I have used 100 epochs to train the model and as I had applied EarlyStopping() function, it can be seen that the training terminated after reaching a validation accuracy of 91.63% with a very low learning rate of 0.0000000005 in order to reach the global minimum.

1. Testing the model:

For testing the model I have again created the ImageDataGenerator() object and took the first 18 images from the test dataframe to plot the images with their corresponding labels according to the trained model. 12 out 18 images were correctly classified. Later, for the sake of this project I have output the images for each image in the dataset.

1. Conclusion and Future Work:

If there were more complex data, this model could have performed better if the model were deeper like the VGG model or if I had used a complex model like RESNET or Inception model which would more appropriately learn the feature of cats and dogs leading to a better accuracy in the testing phase.