**CISC7201 INTRODUCTION TO DATA SCIENCE PROGRAMMING**

**Project for 2019/2020 Semester 1**

**Stanford Dog Recognition**

**Convolution Neural network**

**Transfer Learning Model**

A dog sitting on a blue bench

Description automatically generated

Group A Member:

CHRISTY, CHAN HIO LAM (MB955702)

UN CHON WAI (MB955370)

VONG EUGENE IN CHI (MB955533)

# Topic and Dataset

Our project utilizes a **Convolution Neural network (CNN)** to train and recognize different breeds of dogs. The CNN has a total number of 88 layers, in which 86 layers are hidden layers.

The chosen data set (757MB) is from *Stanford Dogs Dataset*[[1]](#footnote-1). The dataset contains images of 120 breeds of dogs from around the world. This dataset has been built using images and annotation from ImageNet for the task of fine-grained image categorization.



Contents of this dataset:

* Number of categories: 120
* Number of images: 20,580
* Annotations: Class labels, Bounding boxes
* Pixel: 244\*244

70 % of our data set is assigned to be the training set and 30% to be the testing set.

In order to increase the size of our dataset size, our algorithm has implemented **data augmentation** and has doubled the size of our dataset by flipping the images horizontally.

# Libraries

* https://s3.amazonaws.com/keras.io/img/keras-logo-2018-large-1200.png **&** 

“**Keras**”[[2]](#footnote-2) is a high-level neural networks API that provides high-level building blocks for developing deep learning models. And with the features of Keras, it only handles high-level building blocks and does not handle low-level operations such as n-dimension array products, convolutions itself. Instead, Keras relies on a specialized, well optimized tensor manipulation library to do so, serving as the "backend engine" of Keras. Typically, Tensorflow, Theano or CNTK would be used as the backend of a Keras CNN model. Due to the lack of resources in hardware, we have chosen “**PlaidML**[[3]](#footnote-3)” to be the backend engine of our CNN project. PlaidML is an advanced and portable tensor compiler for enabling deep learning on laptops, embedded devices, or other devices where the available computing hardware is not well supported, or the available software stack contains unpalatable license restrictions.

* 

Keras is a deep learning library that derived from **Numpy** library, which is an indispensable library in machine learning as all the image pixels will undergo the process of convolution, pooling and flattening before implementing into the neural network input layer neuron.

* 

**Matplotlib** is a library to visualize our CNN training results, including model learning rate, model loss value and model mean squared error (MSE).

# Convolution Neural network (CNN) – Transfer Learning Model

Transfer learning is a machine learning method where a model developed for a task is retrained as the initial point for a model on a second related task. It is a popular approach in the deep learning area of natural language processing and image vision neural network models development.

The transfer learning model we have used is MobileNetV2[[4]](#footnote-4), a new mobile architecture that improves the performance of mobile models on multiple tasks and benchmarks as well as across a spectrum of different model sizes. It was developed by Google and launched in 2018.

For the learning rate of our CNN model, instead of using a fixed learning rate, we have implemented a gradually descending learning rate which initially sets as 1 x 10-4 and gradually lowers to 1 x 10-8. It provides a better environment for model training and prevents overfitting.

Another strategy to cope with the overfitting problem is adding a dropout layer in our CNN model. With the additional layer, we will set 50% of neurons on a particular layer to be deactivated in the training model. And the implementation of the dropout layer has affirmatively increased the model accuracy in our CNN model.

# Data Collection

Since our data set is downloaded from the website1, 3 functions readDatasetFolderName(), loadDataset () and loadImg() have be used to read the data in our CNN training model.

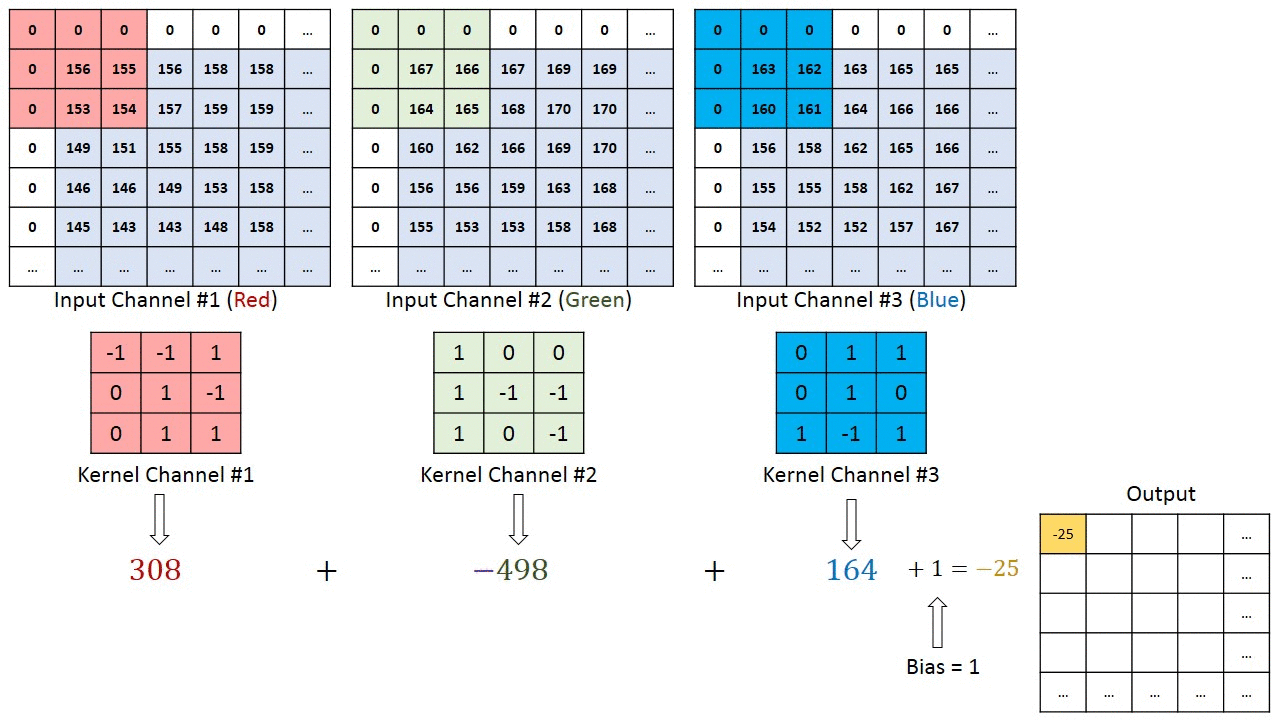
# Data Cleansing

In order to achieve a higher accuracy in our model, data annotation is a mandatory stage for data pre-processing in supervised learning. With the annotations provided along with the dataset (21MB), data annotation technique has been implemented in our CNN training model to cleanse data by removing the background pixel of the images.

# Data Processing

A CNN is a Deep Learning algorithm which can take in an input image, assign importance to various aspects/objects in the image and be able to differentiate one from the other. It is capable to successfully capture the temporal and spatial dependencies in an input image through the application of relevant filters.

A complete CNN model has two more layers comparing to a simple neural network, i.e. Convolutional Layer and Pooling Layer.

1. Convolutional layer: as the kernel shift along the image performing the algorithm like a slicing window, the CNN model will perform a matrix multiplication operation between kernel and the percentage of that color (RGB) of the image over which the kernel is hovering. 

And for our CNN model, the rectified linear unit, i.e. ReLu function = max(0, x), activation function has been used in the Convolutional layer to deliver a faster algorithm comparing to the other activation functions.

1. Pooling Layer: matrix dimension reduction which lowers the need of computational power. The algorithm that we used is Max Pooling, which returns the maximum value from the portion of the image covered by the Kernel.

After implement these two processes, the algorithm is going to flatten the output and step into the neural network.

# Result

After implementing 29 epochs in our CNN model, we uses the Matplotlib library to visualize the result of the Loss Function Value, Model Top Accuracy, Mean Square Error Value and the optimized Learning Rate of our model with the Top 3 accuracy is 94.75%.

A close up of a map

Description automatically generated A picture containing screenshot

Description automatically generated

A screenshot of a social media post

Description automatically generatedA screenshot of a cell phone

Description automatically generated

1. <http://vision.stanford.edu/aditya86/ImageNetDogs/> [↑](#footnote-ref-1)
2. <https://keras.io/> [↑](#footnote-ref-2)
3. <https://github.com/plaidml/plaidml> [↑](#footnote-ref-3)
4. 4. Sandler, M., Howard, A., Menglong, Z., Zhmoginov, A., Chen, L., (2019), MobileNetV2: Inverted Residuals and Linear Bottlenecks [↑](#footnote-ref-4)