Using Neural Networks for Image Recognition

For this portfolio, we will make use of the Logo Images Dataset obtained from https://github.com/msn199959/Logo-2k-plus-Dataset, it contains over 150, 000 images, which are categorised as follows:

Root Category	Logos	Images
Food	769	54,507
Clothes	286	20,413
Institution	238	17,103
Accessories	210	14,569
Transportation	203	14,719
Electronic	191	13,972
Necessities	182	13,205
Cosmetic	115	7,929
Leisure	99	7,338
Medical	48	3,385
Total	2,341	167,140

Introduction

A neural network attempts to replicate the structure of the brain, in which neurons pass electrical current form a directed network. Here if enough neurons sense an the input (for example a touch on the skin), a signal is sent via a charge through the network to produce an output.

An artificial neural network imitates this by having an input layer, hidden layers and an output layer as shown below:

\textbf{REVISIT!!!}

As we can see, the internal neurons can have multiple inputs and outputs. The inputs, $x_1, \ldots, x_n$, have weights $w_1, \ldots, w_n$ and this weighted input is passed to an activation function $\phi()$, to get the output of the neuron as:

$$y = \phi\left(\sum_i x_i w_i\right)$$

A simple neural network is made up of an input layer, hidden layers and an output layers. Our aim is to select correct weights on each edge using iterative methods.

Backpropogation

This is a training method, also referred to "the backward propogation of errors". To use this, we first define the following quatities

$$J(y) = (t-y)^2 \text{ the loss function,} \tag{1}$$

$$D_n(y) = \frac{dJ(y)}{dw_n} \text{ the derivative of the loss function} \tag{2}$$

We then perform the following steps for each $(x, t) \in X$

1. Pass x through the neural network and obtain the output y
2. Obtain the new weight for each edge $w_n' = \delta w_n = -RD_n(y)$ for a learning rate R

The Pet Breed Dataset

In [1]:
```
import pathlib

data_dir = pathlib.Path("Pet_Breeds")
image_count = len(list(data_dir.glob('*/*.jpg')))
print(image_count)
```

```
3712
```

In [2]:
```
import PIL
import PIL.Image

abyssinian = list(data_dir.glob('abyssinian/*'))
PIL.Image.open(str(abyssinian[50]))
```

Out[2]:

In [7]:

```
TF_ENABLE_ONEDNN_OPTS=0
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential

batch_size = 32
img_height = 180
img_width = 180

train_dataset = tf.keras.utils.image_dataset_from_directory(
    data_dir,
    validation_split=0.2,
    subset="training",
    seed=123,
    image_size= (img_height, img_width),
    batch_size= batch_size
)

test_dataset = tf.keras.utils.image_dataset_from_directory(
    data_dir,
    validation_split=0.2,
    subset="validation",
    seed=123,
    image_size= (img_height, img_width),
    batch_size= batch_size
)
```

```
Found 3881 files belonging to 23 classes.
Using 3105 files for training.
Found 3881 files belonging to 23 classes.
Using 776 files for validation.
```

In [10]:

```
class_names = train_dataset.class_names
print(class_names)
```

```
['abyssinian', 'american shorthair', 'beagle', 'boxer', 'bulldog', 'chihuah
ua', 'corgi', 'dachshund', 'german shepherd', 'golden retriever', 'husky',
'labrador', 'maine coon', 'mumbai cat', 'persian cat', 'pomeranian', 'pug',
'ragdoll cat', 'rottwiler', 'shiba inu', 'siamese cat', 'sphynx', 'yorkshir
e terrier']
```

In [9]:

```python
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 10))
for images, labels in train_dataset.take(1):
  for i in range(9):
    ax = plt.subplot(3, 3, i + 1)
    plt.imshow(images[i].numpy().astype("uint8"))
    plt.title(class_names[labels[i]])
    plt.axis("off")
```

```
---------------------------------------------------------------------------
InvalidArgumentError                      Traceback (most recent call last)
Cell In[9], line 4
      1 import matplotlib.pyplot as plt
      3 plt.figure(figsize=(10, 10))
----> 4 for images, labels in train_dataset.take(1):
      5   for i in range(9):
      6     ax = plt.subplot(3, 3, i + 1)

File ~/Documents/COMPASS/First_Year/TB2/Statistical_Computing/tfvenv/lib/python3.8/site-packages/tensorflow/python/data/ops/iterator_ops.py:787, in OwnedIterator.__next__(self)
    785 def __next__(self):
    786   try:
--> 787     return self._next_internal()
    788   except errors.OutOfRangeError:
    789     raise StopIteration

File ~/Documents/COMPASS/First_Year/TB2/Statistical_Computing/tfvenv/lib/python3.8/site-packages/tensorflow/python/data/ops/iterator_ops.py:770, in OwnedIterator._next_internal(self)
    767 # TODO(b/77291417): This runs in sync mode as iterators use an error status
    768 # to communicate that there is no more data to iterate over.
    769 with context.execution_mode(context.SYNC):
--> 770   ret = gen_dataset_ops.iterator_get_next(
    771       self._iterator_resource,
    772       output_types=self._flat_output_types,
    773       output_shapes=self._flat_output_shapes)
    775   try:
    776     # Fast path for the case `self._structure` is not a nested structure.
    777     return self._element_spec._from_compatible_tensor_list(ret)  # pylint: disable=protected-access

File ~/Documents/COMPASS/First_Year/TB2/Statistical_Computing/tfvenv/lib/python3.8/site-packages/tensorflow/python/ops/gen_dataset_ops.py:3017, in iterator_get_next(iterator, output_types, output_shapes, name)
   3015   return _result
   3016 except _core._NotOkStatusException as e:
-> 3017   _ops.raise_from_not_ok_status(e, name)
   3018 except _core._FallbackException:
   3019   pass

File ~/Documents/COMPASS/First_Year/TB2/Statistical_Computing/tfvenv/lib/python3.8/site-packages/tensorflow/python/framework/ops.py:7215, in raise_from_not_ok_status(e, name)
   7213 def raise_from_not_ok_status(e, name):
   7214   e.message += (" name: " + name if name is not None else "")
-> 7215   raise core._status_to_exception(e) from None

InvalidArgumentError: {{function_node __wrapped__IteratorGetNext_output_types_2_device_/job:localhost/replica:0/task:0/device:CPU:0}} Unknown image file format. One of JPEG, PNG, GIF, BMP required.
         [[{{node decode_image/DecodeImage}}]] [Op:IteratorGetNext]
<Figure size 1000x1000 with 0 Axes>
```