Image Classification

Abstract:

Recent advances in deep learning made tasks such as Image and speech recognition possible.

Deep Learning: A subset of Machine Learning Algorithms that is very good at recognizing patterns but typically requires a large number of data. Deep learning excels in recognizing objects in images as it's implemented using 3 or more layers of artificial neural networks where each layer is responsible for extracting one or more feature of the image (more on that later).

Neural Network: A computational model that works in a similar way to the neurons in the human brain. Each neuron takes an input, performs some operations then passes the output to the following neuron.

We're going to teach the computer to recognize images and classify them into one of these 10 categories: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. To do so, we first need to teach the computer how a cat, a dog, a bird, etc. look like before it being able to recognize a new object. The more cats the computer sees, the better it gets in recognizing cats. This is known as supervised learning. We can carry this task by labelling the images, the computer will start recognizing patterns present in cat pictures that are absent from other ones and will start building its own cognition.

We're going to use Python and TensorFlow to write the program. TensorFlow is an open source deep learning framework created by Google that gives developers granular control over each neuron (known as a "node" in TensorFlow) so you can adjust the weights and achieve optimal performance. TensorFlow has many built-in libraries (few of which we'll be using for image classification) and has an amazing community, so you'll be able to find open source implementations for virtually any deep learning topic.

Requirements:

- **Ubuntu:** Ubuntu is one of the best open source software operating system for software developers.

- **Python:** Python is a programming language and the basic code base of our project. The version to be used is Python2.7 as it has the most stable compatible machine learning packages.

- **NumPy:** NumPy is a package in Python used for Scientific Computing. In the experiment NumPy has been used to vectorize the training dataset(sentences).

- **TensorFlow:** TensorFlow is a Python library for fast numerical computing created and released by Google. In our experiment Keras runs on top of TensorFlow.

- **Keras:** Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow.

- **OpenCV:** OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. OpenCV supports the deep learning frameworks TensorFlow, Torch/PyTorch and Caffe.

- **Keras-Applications:** Keras Applications are deep learning models that are made available alongside pre-trained weights. These models can be used for prediction, feature extraction, and fine-tuning. Weights are downloaded automatically when instantiating a model.

    - **VGG19 :** VGG19 model, with weights pre-trained on ImageNet. This model can be built both with 'channels_first' data format (channels, height, width) or 'channels_last' data format (height, width, channels). The default input size for this model is 224x224.
    - **MobileNet:** MobileNet model, with weights pre-trained on ImageNet. Note that this model only supports the data format 'channels_last' (height, width, channels).The default input size for this model is 224x224.

      (*data_format: A string, one of "channels_last" (default) or "channels_first". The ordering of the dimensions in the inputs. "channels_last" corresponds to inputs with shape (batch, steps, channels) (default format for temporal data in Keras) while "channels_first" corresponds to inputs with shape (batch, channels, steps)*)

Approach:

- **Using RGB Values**: Colours could be represented as RGB values (a combination of red, green and blue ranging from 0 to 255). Computers could then extract the RGB value of each pixel and put the result in an array for interpretation. When the computer interprets a new image, it will convert the image to an array by using the same technique, which then compares the patterns of numbers against the already-known objects. The computer then allots confidence scores for each class. The class with the highest confidence score is usually the predicted one.

- **CNN:** One of the most popular techniques used in improving the accuracy of image classification is Convolutional Neural Networks (CNNs for short). Convolutional Neural Network: A special type Neural Networks that works in the same way of a regular neural network except that it has a convolution layer at the beginning. Instead of feeding the entire image as an array of numbers, the image is broken up into a number of tiles, the machine then tries to predict what each tile is. Finally, the computer tries to predict what's in the picture based on the prediction of all the tiles. This allows the computer to parallelize the operations and detect the object regardless of where it is located in the image.

- **Importing the required packages:** VGG19 and MobileNet gives best results.

```
from keras.preprocessing import image as image_utils
from keras.applications.imagenet_utils import decode_predictions
from keras.applications.imagenet_utils import preprocess_input
from keras.applications import VGG19
import numpy as np
import cv2
```

- Load the original image via OpenCV so we can draw on it and display it to our screen later.

- Load the input image using the Keras helper utility while ensuring that the image is resized to 224x224 pixels, the required input dimensions for the network -- then convert the PIL image to a NumPy array.

- Our image is now represented by a NumPy array of shape (224, 224, 3), assuming TensorFlow "channels last" ordering of course, but we need to expand the dimensions to be (1, 3, 224, 224) so we can pass it through the network -- we'll also pre-process the image by subtracting the mean RGB pixel intensity from the ImageNet dataset.

- Finally load the VGG19 network pre-trained on the ImageNet dataset and let the algorithm classify the image.

Scope:

- **Automotive sector:** In developing advanced drivers assist for semi-autonomous cars and also heavily used in autonomous/driver-less cars

- **Image enhancing:** The camera apps in smartphones and digital cameras using image processing to enhance the image quality, video stabilization and noise removal etc.

- **Problem specific solutions:** Image processing is used as a solution to a variety of problems, starting from facial recognition access to defects identification in manufacturing industries

- **Human machine interface:** Machines are made smart by adding gestural interface, or human action response interfaces, which decodes the actions of the human user to perform certain tasks.

- **Medical Sciences:** In recent years the image processing mechanisms are used widely in several medical areas for improving earlier detection and treatment stages, in which the time factor is very important to discover the disease in the patient as possible as fast, especially in various cancer tumours such as the lung cancer.

Conclusion:

We were able to build an artificial convolutional neural network that can recognize images with an accuracy of 78% using TensorFlow and pre-trained weights in VGG19 model.