**Introduction**

*This Project is Based on Harry potter and its characters:*

*You Have to pass an image as a URL and it will generate an name of the character written as caption on the image.*

*Characters to be used as input Data:*

1. *Albus Dumbledore*
2. *Draco Malfoy*
3. *Harry Potter*
4. *Lord Voldemort*
5. *Hermione Granger*
6. *Severus Snape*

**History**

*A few years ago, the creation of the software and hardware image processing systems was mainly*

*limited to the development of the user interface, which most of the programmers of each firm were*

*engaged in. The situation has been significantly changed with the advent of the Windows operating*

*system when the majority of the developers switched to solving the problems of image processing*

*itself. However, this has not yet led to the cardinal progress in solving typical tasks of recognizing*

*faces, car numbers, road signs, analysing remote and medical images, etc. Each of these "eternal"*

*problems is solved by trial and error by the efforts of numerous groups of the engineers and scientists.*

*As modern technical solutions are turn out to be excessively expensive, the task of automating the*

*creation of the software tools for solving intellectual problems is formulated and intensively solved*

*abroad. In the field of image processing, the required tool kit should be supporting the analysis and*

*recognition of images of previously unknown content and ensure the effective development of*

*applications by ordinary programmers. Just as the Windows toolkit supports the creation of interfaces*

*for solving various applied problems.*

*Now a days various frameworks are available to increase the accuracy and calculation power with their optimised prebuilt models such as* ***fastai****,* ***Keras*** *etc.*

**Objective**

The aim of object detection is to detect all instances of objects from a known class, such as people,

cars or faces in an image. Generally, only a small number of instances of the object are present in the

image, but there is a very large number of possible locations and scales at which they can occur and

that need to somehow be explored. Each detection of the image is reported with some form of pose information. This is as simple as the location of the object, a location and scale, or the extent of the object defined in terms of a bounding box. In some other situations, the pose information is more

detailed and contains the parameters of a linear or non-linear transformation. For example for face

detection in a face detector may compute the locations of the eyes, nose and mouth, in addition to the

bounding box of the face. An example of a bicycle detection in an image that specifies the locations

of certain parts. The pose can also be defined by a three-dimensional transformation specifying the location of the object relative to the camera. Object detection systems always construct a model for an object class from a set of training examples. In the case of a fixed rigid object in an image, only one example may be needed, but more generally multiple training examples are necessary to capture certain aspects of class variability.

**Framework(Fastai)**

fastai is a deep learning library which provides practitioners with high-level components that can quickly and easily provide state-of-the-art results in standard deep learning domains, and provides researchers with low-level components that can be mixed and matched to build new approaches. It aims to do both things without substantial compromises in ease of use, flexibility, or performance. This is possible thanks to a carefully layered architecture, which expresses common underlying patterns of many deep learning and data processing techniques in terms of decoupled abstractions. These abstractions can be expressed concisely and clearly by leveraging the dynamism of the underlying [Python](https://www.python.org/) language and the flexibility of the [PyTorch](https://pytorch.org/) library. fastai includes:

* A new type dispatch system for Python along with a semantic type hierarchy for tensors
* A GPU-optimized computer vision library which can be extended in pure Python
* An optimizer which refactors out the common functionality of modern optimizers into two basic pieces, allowing optimization algorithms to be implemented in 4-5 lines of code
* A novel 2-way callback system that can access any part of the data, model, or optimizer and change it at any point during training
* A new data block API
* ...and much more.

We have used this library to successfully create a complete deep learning course, which we were able to write more quickly than using previous approaches, and the code was more clear. The library is already in wide use in research, industry, and teaching.

For further Reference : <https://www.fast.ai/2020/02/13/fastai-A-Layered-API-for-Deep-Learning/#:~:text=Abstract%3A%20fastai%20is%20a%20deep,matched%20to%20build%20new%20approaches.>

Figure

**Existing Method**

**Resnset :**

This is the Training model used in the given project. It is basically categorised into subcategories based on the number of neural networks present in the training model such as 51,101 etc.

It is one of the most efficient ways to train our model, it is prebuilt in the fastai framework that we have used to prepare our data for the training purpose.

**R-CNN:**

CNN

Another type of Training Model used for training the data. To circumvent the problem of selecting a huge number of regions, Ross Girshick et al. proposed a method where we use the selective search for extract just 2000 regions from the image and he called them region proposals. Therefore, instead of trying to classify the huge number of regions, you can just work with 2000 regions.

**System Requirement**

We have used the Notebook Provided by the google, i.e Google Colab. It is linux based environment provided by the google free of cost. The google also provides very fast severs to download the dataset as well free GPU and TPU to boost the time required to train the working model.

Libraries Used:

1. Fastai
2. Pandas
3. openCV

Steps To be followed for installing the Above Libraries,

Install the following dependencies via pip:

1 . !pip install fastai(‘!’ used in google colab only )

2 . !pip install pandas(‘!’ used in google colab only)

3 . !pip install openCV(‘!’ used in google colab notebook only)

**RESULTS**

**Before Detection: After detection:**

** **