**BE Capstone Project Report**

**IMAGE UPLOAD WEB APP WITH OBJECT DETECTION**

By- Shivam Agarwal Shubham Jha

Satyajeet Salhunke Sanjeet Jha

### Project Guide :- Prof. Prakash Waghamode

**INDEX**

| SR No. | Title | Page no. |
| --- | --- | --- |
| 1. | Cover Page | 1 |
| 2. | Acknowledgement | 2 |
| 3. | Introduction | 4 |
| 4. | Literature Survey | 6 |
| 5. | Approach | 11 |
| 6. | Project requirements | 20 |
| 7. | Results | 23 |
| 8. | Research Gap | 25 |
| 9. | Project Plan | 26 |
| 10. | Conclusion | 27 |
| 11. | References | 28 |

**Introduction**

More than 90% of adults now have mobile phones most of which allow them to take photos and instantly store and share them on online platforms. Half of all Internet users (54%) share original photos and videos online, and an increasing number are using specific photo storing applications

User studies have started to explore what type of content is being shared, categories of users, and the role of social features within a photo-based social network .

Image upload webapp is a photo sharing , object detection and classification and storage service.

The service will be free and unlimited. The service automatically analyzes photos, identifying various visual features and classes of objects.

users can search for a photo, with the service returning results from categories such as People, Places, birds, animals, aeroplane, bicycle etc.

The computer vision of photos recognizes classes of objects ,grouping similar one together and subject matters including buildings, animals, food and more.

Humans glance at an image and instantly know what objects are in the image, where they are, and how they interact. The human visual system is fast and accurate, allowing us to perform complex tasks . Fast, accurate algorithms for object detection would allow computers to drive cars without specialized sensors, enable assistive devices to convey real-time scene information to human users, and unlock the potential for general purpose, responsive robotic systems.

We reframe object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities. Recognizing shapes is a cool and easy example, but obviously it’s not what we do in the real world. Also, our algorithm can only predict a fixed number of bounding boxes per image. In the real world, however, we have diverse scenarios: A small side road may have no cars on it, but as soon as we drive on the highway, we have to recognize hundreds of cars at the same time. Even though this seems like a minor issue, it’s actually very hard to solve . It’s the algorithm that decides what’s an object and what’s background, but it can get confused if it doesn’t know how many objects are there in the image.

When we look at a tree from close by, Even though we only see a bunch of leaves and sticks, we can still clearly say it’s all one object, because our mind understands what a tree is. If the leaves were lying around on the floor instead, we would easily detect them as individual objects. Unfortunately, neural networks don’t quite understand what trees are, so this is a pretty hard challenge for them. So that’s why they need to be trained more number of times with specific datasets with exactly the class of objects they need to detect in order for most accurate predictions.

We have many Deep neural network algorithms for doing this object detection tasks, such as RCNN, Fast RCNN, Faster RCNN, and CNN in general for classification purposes.

**Literature Survey**

## Model experiment

### To provide an experiment and compare results with keys for each plant, 500 photos from online-classifier “The list of plants of the Dneprovskiy district of Kiev” (Fig.

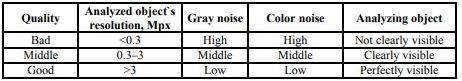
2) were taken. The online-classifier contains the pictures of each kind of the plants and its determination names. Photos were characterized by the method described in

3.2 due to the different quality of the photos and collected.



**The general method of photo analysis**

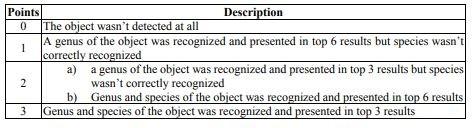
Photo’s quality is an important factor to Google Photos. Therefore, it is necessary to classify each photo by main quality components – composition, resolution, digital noise. Main photos quality criteria are presented in table .



**Data collection and analysis**

To collect data, we developed the database with front-end and back-end development. Each photo was classified by the image quality using the method described , and its characteristics such as type (tree, bush, grass) and presented part of the plant (flower, leaf, stem, fruit). The mark of the analyzing process was inputted too.

The output interface looked like a table to provide the visualization and dynamic of the research process. Google proposed a few results of the analysis to the user. Therefore, the results were classified on 0, 1, 2 or 3 points. Sometimes cropping of the photos was used, in this case, one point was deducted.



Results were collected on the database. To provide an analysis of the requests to a database prepared and provided. The requests were prepared to take into account the aims of the work. To process the results MS Excel 365 was used.

**Search by image to reveal copies of a known image**

A more recent type of searching, which deals also mainly with images, is search by image or “reverse image search” as it is named by the leading search system that is provided by Google. In this type of searching, a query consists not of a word or a combination of words, but of one image; this image is of course already known by the user who hopes to find images that are relevant in the context of the user’s information need. The search system can retrieve mainly other images with similar distributions of picture elements. Therefore such a search can reveal mainly so-called “copies” of the image that was used in the query.

This method can be quite useful to detect plagiarism. More generally, this type of search allows us to determine reuse of an image that has been available / shared / published on the Internet; this is welcomed for instance by managers of digital libraries that include images (see for instance [5, 9]). Finding images that are semantically similar may be thought of as another obvious application, but this is hindered by the so-called semantic gap, as explained further

#### Evaluation of the automatic classifications in Albums

The quality of the automatic classifications / annotations / tags was evaluated as follows. For each Album with corresponding simple name / annotation / tag, that has been automatically created by Google, each image that has been classified in that category / class was inspected; then it was evaluated if the classification made sense,

i.e. was correct or not. In other words, a bimodal model was used. This is a simple, rudimentary model as the quality of a classification has many aspects / dimensions and can be assigned many levels between correct or not, 1 or 0. This approach is justified in view of the modest aims and size of this case study.

**Object Detection:**

Object detection is a computer vision technique for locating instances of objects in images or videos. Object detection algorithms typically leverage machine learning or deep learning to produce meaningful results. When humans look at images or video, we can recognize and locate objects of interest within a matter of moments. The goal of object detection is to replicate this intelligence using a computer.Object detection is a key technology behind advanced driver

assistance systems (ADAS) that enable cars to detect driving lanes or perform pedestrian detection to improve road safety. Object detection is also useful in applications such as video surveillance or image retrieval systems.

**APPROACH**

**(For Object Detection)**

**What is Object Recognition?**

Object recognition is a general term to describe a collection of related computer vision tasks that involve identifying objects in digital photographs.

### Image classification involves predicting the class of one object in an image. Object localization refers to identifying the location of one or more objects in an image and drawing a bounding box around their extent. Object detection combines these two tasks and localizes and classifies one or more objects in an image.

When a user or practitioner refers to “object recognition“, they often mean “object detection“.

Image Classification: Predict the type or class of an object in an image.

Input: An image with a single object, such as a photograph.

Output: A class label (e.g. one or more integers that are mapped to class labels).

Object Localization: Locate the presence of objects in an image and indicate their location with a bounding box.

Input: An image with one or more objects, such as a photograph.

Output: One or more bounding boxes (e.g. defined by a point, width, and height).

Object Detection: Locate the presence of objects with a bounding box and types or classes of the located objects in an image.

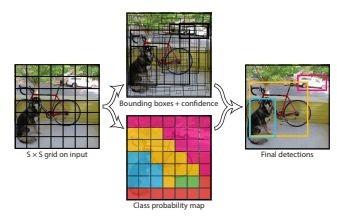
Input: An image with one or more objects, such as a photograph.

Output: One or more bounding boxes (e.g. defined by a point, width, and height), and a class label for each bounding box.

One further extension to this breakdown of computer vision tasks is object segmentation, also called “object instance segmentation” or “semantic segmentation,” where instances of

recognized objects are indicated by highlighting the specific pixels of the object instead of a coarse bounding box.

From this breakdown, we can see that object recognition refers to a suite of challenging computer vision tasks.



Overview of Object Recognition Computer Vision Tasks

Most of the recent innovations in image recognition problems have come as part of participation in the ILSVRC tasks.

This is an annual academic competition with a separate challenge for each of these three problem

types, with the intent of fostering independent and separate improvements at each level that can be leveraged more broadly. For example, see the list of the three corresponding task types below taken from the 2015 ILSVRC review paper:

Image classification: Algorithms produce a list of object categories present in the image. Single-object localization: Algorithms produce a list of object categories present in the image,

along with an axis-aligned bounding box indicating the position and scale of one instance of each

object category.

Object detection: Algorithms produce a list of object categories present in the image along with an axis-aligned bounding box indicating the position and scale of every instance of each object category.

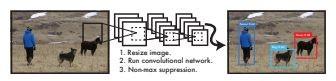
We can see that “Single-object localization” is a simpler version of the more broadly defined

“Object Localization,” constraining the localization tasks to objects of one type within an image, which we may assume is an easier task.

Below is an example comparing single object localization and object detection, taken from the ILSVRC paper. Note the difference in ground truth expectations in each case.

Comparison Between Single Object Localization and Object Detection.

The performance of a model for image classification is evaluated using the mean classification error across the predicted class labels. The performance of a model for single-object localization is evaluated using the distance between the expected and predicted bounding box for the

expected class. Whereas the performance of a model for object recognition is evaluated using the precision and recall across each of the best matching bounding boxes for the known objects in the image.

Now that we are familiar with the problem of object localization and detection, let’s take a look at some recent top-performing deep learning models.

* 1. NN Model Family

The R-CNN family of methods refers to the R-CNN, which may stand for “Regions with CNN Features” or “Region-Based Convolutional Neural Network,” developed by Ross Girshick, et al.

This includes the techniques R-CNN, Fast R-CNN, and Faster-RCNN designed and demonstrated for object localization and object recognition.

**R-CNN**

It is one of the first large and successful application of convolutional neural networks to the problem of object localization, detection, and segmentation. The approach was demonstrated on benchmark datasets, achieving then state-of-the-art results on the VOC-2012 dataset and the 200-class ILSVRC-2013 object detection dataset.

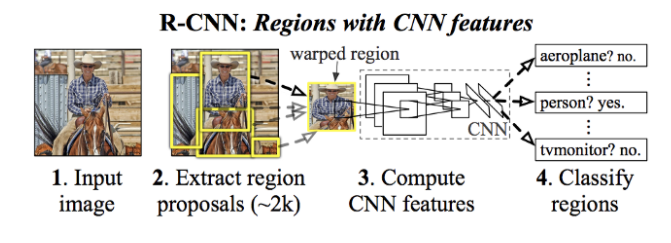
Their proposed R-CNN model is comprised of three modules; they are:

Module 1: Region Proposal. Generate and extract category independent region proposals, e.g. candidate bounding boxes.

Module 2: Feature Extractor. Extract feature from each candidate region, e.g. using a deep convolutional neural network.

Module 3: Classifier. Classify features as one of the known class, e.g. linear SVM classifier model.

A computer vision technique is used to propose candidate regions or bounding boxes of potential objects in the image called “selective search,” although the flexibility of the design allows other region proposal algorithms to be used.



To bypass the problem of selecting a huge number of regions, [Ross Girshick et al](https://arxiv.org/pdf/1311.2524.pdf). proposed method where we use selective search to extract just 2000 regions from the image and he called them region proposals. Therefore, now, instead of trying to classify a huge number of regions, you can just work with 2000 regions. These 2000 region proposals are generated using the selective search algorithm.

Selective Search:  
1. Generate initial sub-segmentation, we generate many candidate regions  
2. Use greedy algorithm to recursively combine similar regions into larger ones   
3. Use the generated regions to produce the final candidate region proposals

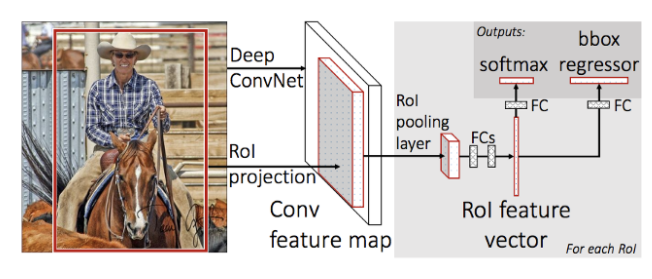
The feature extractor used by the model was the AlexNet deep CNN that won the ILSVRC-2012 image classification competition. The output of the CNN was a 4,096 element vector that describes the contents of the image that is fed to a linear SVM for classification, specifically one SVM is trained for each known class.

It is a relatively simple and straightforward application of CNNs to the problem of object localization and recognition. A downside of the approach is that it is slow, requiring a

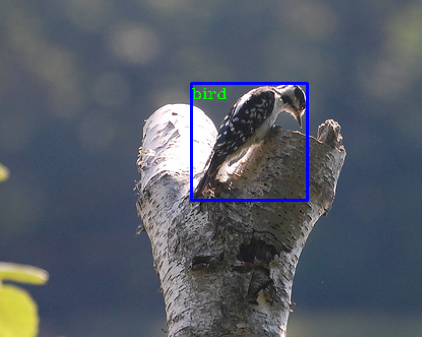
CNN-based feature extraction pass on each of the candidate regions generated by the region proposal algorithm. This is a problem as the paper describes the model operating upon

approximately 2,000 proposed regions per image at test-time.

**Fast R-CNN**

****

To solve some of the drawbacks of R-CNN and to build a faster object detection algorithm Fast R-CNN was introduced. The approach is similar to the R-CNN algorithm. But, instead of feeding the region proposals to the CNN, we feed the input image to the CNN to generate a convolutional feature map. From the convolutional feature map, we identify the region of proposals and warp them into squares and by using a RoI pooling layer we reshape them into a fixed size so that it can be fed into a fully connected layer. From the RoI feature vector, we use a softmax layer to predict the class of the proposed region and also the offset values for the bounding box.The reason “Fast R-CNN” is faster than R-CNN is because we don’t have to feed 2000 region proposals to the convolutional neural network every time. Instead, the convolution operation is done only once per image and a feature map is generated from it.



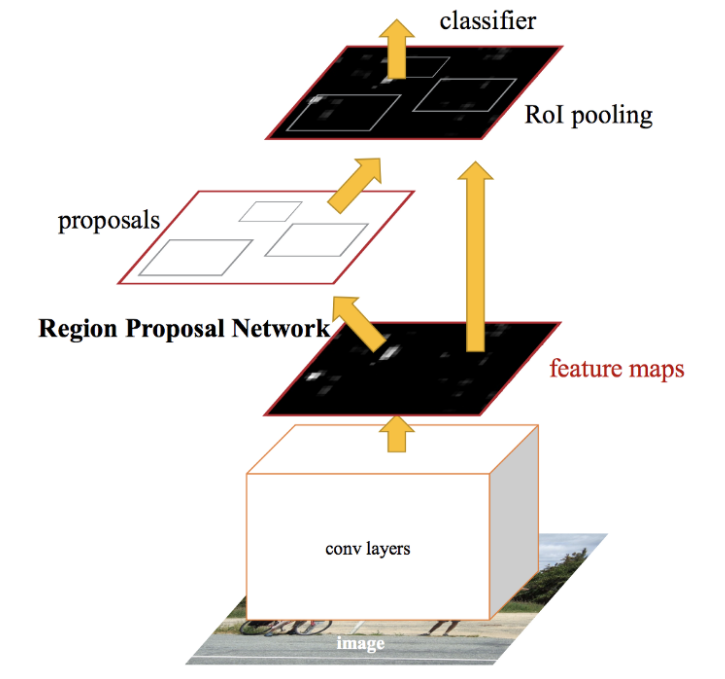
Training a deep CNN on so many region proposals per image is very slow.Object detection is slow. Make predictions using a deep CNN on so many region proposals is very slow.

Fast R-CNN is proposed as a single model instead of a pipeline to learn and output regions and classifications directly.

The architecture of the model takes the photograph a set of region proposals as input that are passed through a deep convolutional neural network. A pre-trained CNN, such as a VGG-16, is used for feature extraction. The end of the deep CNN is a custom layer called a Region of

Interest Pooling Layer, or RoI Pooling, that extracts features specific for a given input candidate region.

**Faster R-CNN**



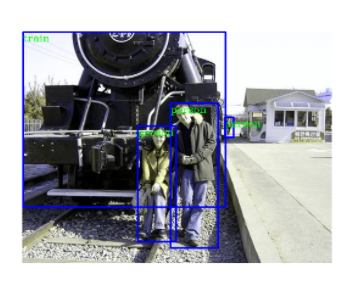
Selective search is a slow and time-consuming process affecting the performance of the network. Therefore, a new object detection algorithm was designed that eliminates the selective search algorithm and lets the network learn the region proposals. Similar to Fast R-CNN, the image is provided as an input to a convolutional network which provides a convolutional feature map.

Instead of using selective search algorithm on the feature map to identify the region proposals, a separate network is used to predict the region proposals. The predicted region proposals are then reshaped using a RoI pooling layer which is then used to classify the image within the proposed region and predict the offset values for the bounding boxes.

Faster R-CNN is much faster than its predecessors , therefore it can be used for real time object Detection.

The steps followed by Faster R-CNN algorithm to detect objects in an image:

1. We take an input image and pass it to the ConvNet which returns feature maps for the image.
2. Then apply Region Proposal Network (RPN) on these feature maps and get object proposals
3. Then apply ROI pooling layer to bring down all the proposals to the same size.
4. Finally, pass these proposals to a fully connected layer in order to classify any predict the bounding boxes for the image.



**Problem Statement**

#### Project Scope

* + 1. This website provides an image upload Web Service which detects classes within images and automatically group them into albums.
    2. It has a search engine which shows images of the text query.
    3. Using Amazon S3 buckets for storing user images.

#### Project Assumptions

Made regarding budgeting , scheduling constraints, and resource . Assumptions made about these factors are circumstances that are presumed to be true in the future, and project strategies are built around them.

1. Resource Assumption
2. Datasets will be taken from Kaggle or compiled using a python script
3. Next we will use these dataset to train our model

#### Technology assumption

We are using Faster R-CNN object detection algorithm for training our dataset and detect the classes of objects in the image.

#### Cost assumption.

All the softwares we are using is Open Source. The only cost that can arise is due to Amazon Aws Hosting on Ec2 and Kubernetes Engine.

* 1. **Time-based assumptions**

We have divided the Project in 3-3 months,first 3 months we will do project planning & designing and next 3 months will do project implementation.

#### Project Limitation:-

We are not able to find all the datasets on Kaggle and other sources, for that we will need to use a script which downloads images from Google images. Since it's not labelled we will need to manually create labels which is a very lengthy process. Also since we can’t consider all the classes of images because that will be an enormous amount , the model detects limited classes of images on which it is trained.

#### Project Objective:-

It is Web App which automatically detects objects in an image and creates an album for the same.When the user will search for an Image using a class name the model will automatically detect the image and will display the images of that class.

.

**Project Requirements**

## Resources

#### Human Resources

* + 1. 4 persons(Team Members)

#### Reusable Software Components :- NA

* 1. **Software requirements**

3.1. S/W:-

* 1. Google Colab
  2. VS code
  3. NodeJS
  4. ReactJS
  5. Flask (Python)
  6. Tesseract(OCR)
  7. Faster R-CNN Algorithm

#### H/W requirements

* + 1. Any Operating System like Ubuntu
    2. 4 GB Ram
    3. 2GHZ processor
    4. GPU

#### Requirements Rationale

| **Requirements** | **Rationale** |
| --- | --- |
| Functional Requirements | System tasks functions such as taking image input and then processing it into detection |
| Performance Requirements | The Objected Detected should have a high accuracy |
| Usability Requirements | The frontend will be very user-friendly |
| Interface Requirements | Website |
| Operational Requirements | Datasets will be compiled manually and from Kaggle |
| Design Constraints | System is complex but yet our design while taking considerations of different aspects is  simple. |
| Cost and Schedule Constraints | All software are open source,but amazon aws  hosting there can be charges for extra resources used. |

[Table 1.Requirement Rationale]

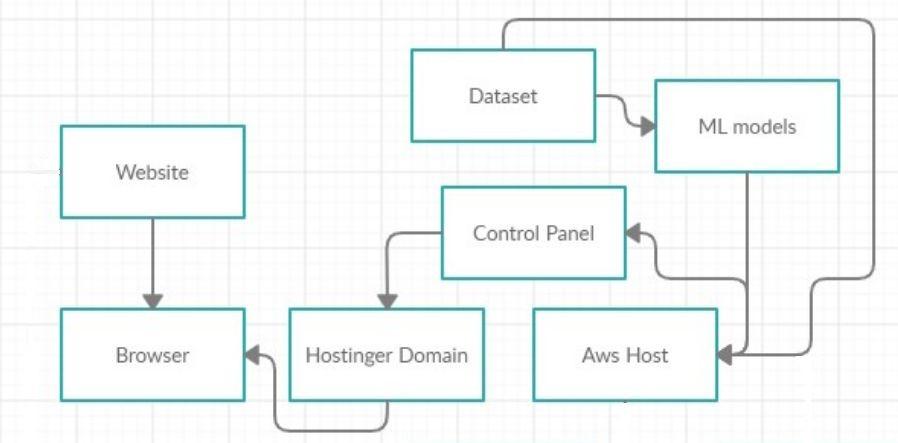
#### Risk Management

**3.1.** Project Risk factors in Table format

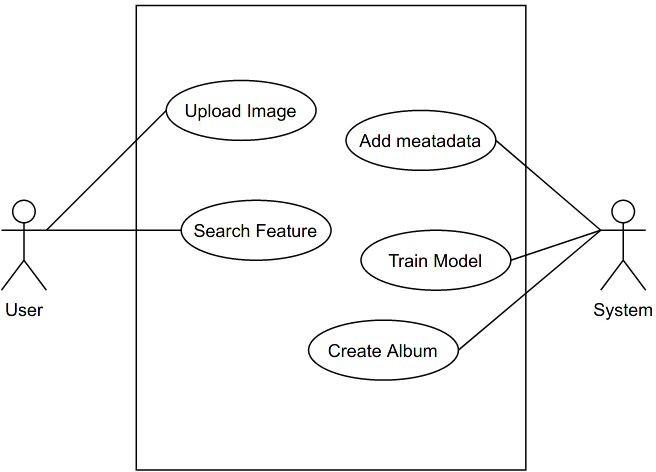
| **Low Risk** | | | | **Medium Risk** | **High Risk** |
| --- | --- | --- | --- | --- | --- |
| Scalability | is | not | implemented | Object Detection can’t be | Albums are not being |
| properly |  |  |  | precise and not detect any  objects or return incorrect | created automatically.And the model is not giving correct image when searched for it |
|  |  |  |  | results |  |

Table 2. Project Risk

#### System Analysis Proposed Architecture/ high level design of the project



[Fig 2. Website Architecture ]



[Fig 3. Use Case Diagram]

# Results

We created a Login page layout on which the user has to fill login credentials in order to log in to their accounts. After successful login the user will enter into its account and the dashboard will be displayed. Here user can upload , view previously uploaded photos and search for particular images.

Our Data set consists 20 different classes of images labeled in Pascal VOC format. The dataset has around 39500 images ,all labeled.

We used Faster R-CNN Algorithm to train our model on the dataset.

Bounding boxes have been created according to the classes that we have created.

**Research Gap**

The first size challenge is that for users with large photo collections there is simply too much metadata. Sending even minimal information (the photo urls, width, height, and timestamps) is many megabytes of data for a full collection, this would directly run counter to our goal of near-instant loading.

The second size challenge is the photos themselves. With modern HDPI screens, even a small photo thumbnail is often 50KB or more. A thousand thumbnails may be 50 megabytes, and not only is that a lot to download, if you try to place them all in the webpage immediately you can slow down the browser. The old Google+ Photos would become sluggish after scrolling through 1000–2000 photos and Chrome would eventually crash the tab after loading 10000.

Scrubbable Photos — the ability to quickly jump to any part of the photo library. Justified Layout — fill the width of the browser and preserve the aspect-ratio of each photo (no square crops).

60fps Scrolling — ensuring the page remains responsive even when looking at many thousands of photos.

Instantaneous Feel —minimize the time waiting for anything to load.

**Project Plan**

* First of all we researched the problem statement and did an in depth analysis of the topic.
* We read Different research papers of google photos ,google lens , numerous case studies of the photo sharing and storing webapps and different algorithms.
* We did research on CNN, R-CNN, Fast rcnn , Faster Rcnn and how to train our model by using a labeled dataset and then check the results.
* We have to take care of both front end and back end working simultaneously.
* **So we will be creating a Login Page in the beginning using React.js templates.**
* **Users can enter their credentials and then login into their account.**
* **Then we are creating a User friendly dashboard on which user can view the uploaded photos and also upload new photos.**
* **The photos that user uploads will be efficiently managed by Amazon S3 bucket on AWS.The images will be end to end Encrypted. The idea here is for the user to have its own private cloud which will be managed by the user himself with no interference from 3rd parties.**
* For the classification of the photos and the image class identification We will train the model using Faster R-CNN Algorithm for real time object Detection , based on Different classes of dataset(people, buildings, trees, pets and more).

| Planning | Sep- Oct 2020 |
| --- | --- |
| Analysis | Nov -Dec 2020 |
| Designing | Jan - Feb 2021 |
| Implementation | March- May2021 |
| Deployment | June 2021 |

**Conclusion**

Thus, We have found a very accurate and efficient way to train our dataset for object detection and create a model that can accurately detect the classes of the objects in an image. We will be using the Flask python framework in the Backend for doing all the tasks and also to connect our Faster R-CNN model to the webapp.

For the frontend part we used React.Js templates for web app creation and also for creating the dashboard.

**References**

* H. Jiang and E. Learned-Miller, "Face Detection with the Faster R-CNN," 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017), 2017, pp. 650-657, doi: 10.1109/FG.2017.82.'

### Sun, X., Wu, P. and Hoi, S.C., 2018. Face detection using deep learning: An improved faster RCNN approach. Neurocomputing, 299, pp.42-50.

* Chen, X. and Gupta, A., 2017. An implementation of faster rcnn with study for region sampling. arXiv preprint arXiv:1702.02138.
* Smith, R., 2007, September. An overview of the Tesseract OCR engine. In Ninth international conference on document analysis and recognition (ICDAR 2007) (Vol. 2, pp. 629-633). IEEE.
* Patel, C., Patel, A. and Patel, D., 2012. Optical character recognition by open source OCR tool tesseract: A case study. International Journal of Computer Applications, 55(10), pp.50-56
* Mori, S., Suen, C.Y. and Yamamoto, K., 1992. Historical review of OCR research and development. Proceedings of the IEEE, 80(7), pp.1029-1058.
* Zhou, J., Lopresti, D.P. and Lei, Z., 1997, April. OCR for world wide web images. In Document Recognition IV (Vol. 3027, pp. 58-66). International Society for Optics and Photonics.
* Nieuwenhuysen, P., 2018, February. Information Discovery and Images A Case Study of Google Photos. In 2018 5th International Symposium on Emerging Trends and Technologies in Libraries and Information Services (ETTLIS) (pp. 16-21). IEEE.
* Cai, Z. and Vasconcelos, N., 2018. Cascade r-cnn: Delving into high quality object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 6154-6162).
* Rao, N.V., Sastry, A.S.C.S., Chakravarthy, A.S.N. and Kalyanchakravarthi, P., 2016. OPTICAL CHARACTER RECOGNITION TECHNIQUE ALGORITHMS. Journal of Theoretical & Applied Information Technology, 83(2).