A REPORT

ON

‘DEVELOPING A CONTENT BASED IMAGE RETRIEVAL MODEL’

BY

Name of the Student	ID Number
AVTANSH PANDEY	**2017AAPS0368G**

AT

MapmyIndia, Bengaluru

A Practice School–II Station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

December, 2020

A REPORT
ON

‘DEVELOPING A CONTENT BASED IMAGE RETRIEVAL MODEL’

BY

Name of the Student	ID No.	Discipline
AVTANSH PANDEY	**2017AAPS0368G**	**B.E.(Hons.) Electronics and Communication**

Prepared in partial fulfillment of the
Practice School-II Course No. BITS F412
AT
MapmyIndia, Bengaluru
A Practice School–II Station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
December, 2020

ACKNOWLEDGEMENTS

I would like to thank BITS Practice School Division for this excellent opportunity to pursue my Practice School-II program at MapmyIndia, Bengaluru.

I would like to express my heartfelt gratitude towards Mr. Rakesh Verma, MD Chairman, MapmyIndia, and Mr. Rohan Verma, CEO, MapmyIndia, for giving me an opportunity to be part of an elite organization and contribute to the development of the CBIR Model.

I would also like to thank Mr. Muraleedhara Navada, Principal Engineer, for facilitating the process and for being my project who was not only overseeing the project personally, but also guided me whenever I was in need of any help.

I am also grateful to my PS Faculty, Mr. Ashish Narang, for his dedication towards students currently in the PS and for his quick response to all our queries.

Last but not the least, I would like to thank my parents, A.K. Pandey and Vinita Pandey, for their immense support throughout the project.

Without the help and contribution of the above-mentioned people, this project would not have been a success.

ABSTRACT SHEET

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCES PILANI (RAJASTHAN)

Practice School Division

Station: MapmyIndia	**Centre**: Bengaluru
Duration: 5 months	**Date of Start**: August 6, 2020
Date of Submission: November 30, 2020	

Title of the Project: "Developing a Content Based Image Retrieval Model".

ID No: 2017AAPS0368G	**Name:** AVTANSH PANDEY

Discipline: B.E (Hons.) Electronics and Communication

Name and Designation of the Expert: Mr. Muraleedhara Navada,
Principal Engineer

Name of the PS Faculty: Mr. Ashish Narang

Key Words: CBIR, Scripting, Deep Learning, CNN, Mask R-CNN, Database, Neural Networks, Map Generation
Project Areas: Computer Vision, Machine Learning

Abstract: The current map generation model leads to an accuracy ranging from 30-80% and our aim is to improve the performance of it by developing and implementing a Content Based Image Retrieval Model and re-annotating the retrieved images which are then added to the dataset and retrained on the company's map generation architecture, thus resulting into improvement in overall performance of it.

[illegible]

Signature of Student	Signature of Faculty
Date: October 4, 2020	Date: October 4, 2020

TABLE OF CONTENTS

1 Introduction

Maps are an essential tool in our day to day life. It's utility is not just restricted to navigation but also to something as important as familiarizing with our surroundings and locality. MapmyIndia has developed maps to provide the audience with minute details of their surroundings with various additional features in their application. With advancement in technologies, MapmyIndia has adapted Artificial Intelligence to generate their maps, and for more real-time visuals, they use the Terrestrial Gaze to create their optimized image dataset and develop HD and 3D maps.

1.1 About the organization

MapmyIndia has created a niche, bringing India on the global-map of navigation and telematics. The company, founded over 22 years ago, has built comprehensive, accurate and feature enriched map datasets. These maps are professionally curated, continuously updated and used in navigation, telematics, analytics, and location based services, and have been adapting the latest technological advancements for their products. These maps are enabled on the Internet of Things platforms and mobile applications, and are being used extensively in Smart City projects, Electric and Autonomous Vehiclesand many more disruptive technologies.

1.2 MapmyIndia Teams

1.2.1 Artificial Intelligence

The AI team works on the generation and detection of various aspects of their HD and 3D maps for example, junction detection, road signs, signals, scene detections, etc. It also focuses on the devops and the architecture of the maps with the development of their 3 Box model which serves as the backbone for the same. The team is also responsible for creating structural pipelines for the generation of 3D AI maps and using annotating tools

1.2.2 Tooling

The Tooling team focuses on creating various components and vectors of the maps to be used for further generation. Various tools like MazePCD, MazeEditor, MazeCBR etc., have been created for purposes like digitizing and creating predictions, vector tiles and also on Gaze Picture Referencing and HD Map formatting.

1.2.3 Content

The Content team is responsible for the map data. They also work Map editing, PCD digitization and map annotation which is further used by the other two for processing the maps.

1.3 My Team and Role

I am part of the Artificial Intelligence team and am responsible for implementing a CBIR model that improves the overall performance of the predictions in the 3 Box model which serves as the backbone architecture of MapmyIndia’s HD and 3D maps. The model implementation requires planning and developing a new architecture involving tasks like scripting, re-annotating, image retrieval, database generation, to be deployed parallely with the current architecture.

2 Problem Statement

MapmyIndia has developed a model to generate maps and its features using Computer Vision techniques and they use a Terrestrial Gaze to create their optimized image dataset. Their current 3-Box Model in the system provides an accuracy ranging from 30-80%. Hence we need to devise and implement a method to improve the overall performance of the system using the image queries causing errors. Content Based Image Retrieval (CBIR) technique is to be modelled to retrieve similar images from the developed Feature Database. The need arises from the fact that the maps, which is even used for navigation purposes, cannot afford any errors hence requires various techniques to achieve highest possible accuracies. Therefore, CBIR being one of the most promising techniques is being implemented to improve the overall performance of the model.

GitHub Link: https://github.com/avtanshpandey/CBIR_Deep-Learning

2.1 Project Plan

Step 1: Acquire the dataset of images with annotations/ground truth values and corresponding predictions generated from the ODS model.

Step 2: Divide the dataset into two parts (for eg: 9:1 ratio), the larger one being the Haystack (H) and the smaller one being Reference (R).

Step 3: Extract the feature vectors of the Haystack (H) dataset using the PyTorch implemented model and store it in a database.

Step 4: The Reference dataset is used to identify the false positives in it and using them as query images, call it Needle (N).

Step 5: Extract the features of Needles (N) and use it as the input queries to retrieve similar images from the Haystack (H) using the CBIR model. These retrieved images will be called the shortlisted images (S).

Step 6: Haystack (H) will have the mAP evaluated with respect to the ODS model. Similarly evaluate the mAP for the following subsets:-

- H-S - Haystack without the retrieved images with false positive ground truth values.
- S - Shortlisted images which are retrieved from the Haystack performing CBIR using the Needle queries and have false positive ground truth values like the queries.

Step 7: The mAP values should be in the following order relatively: H-S > H > S.

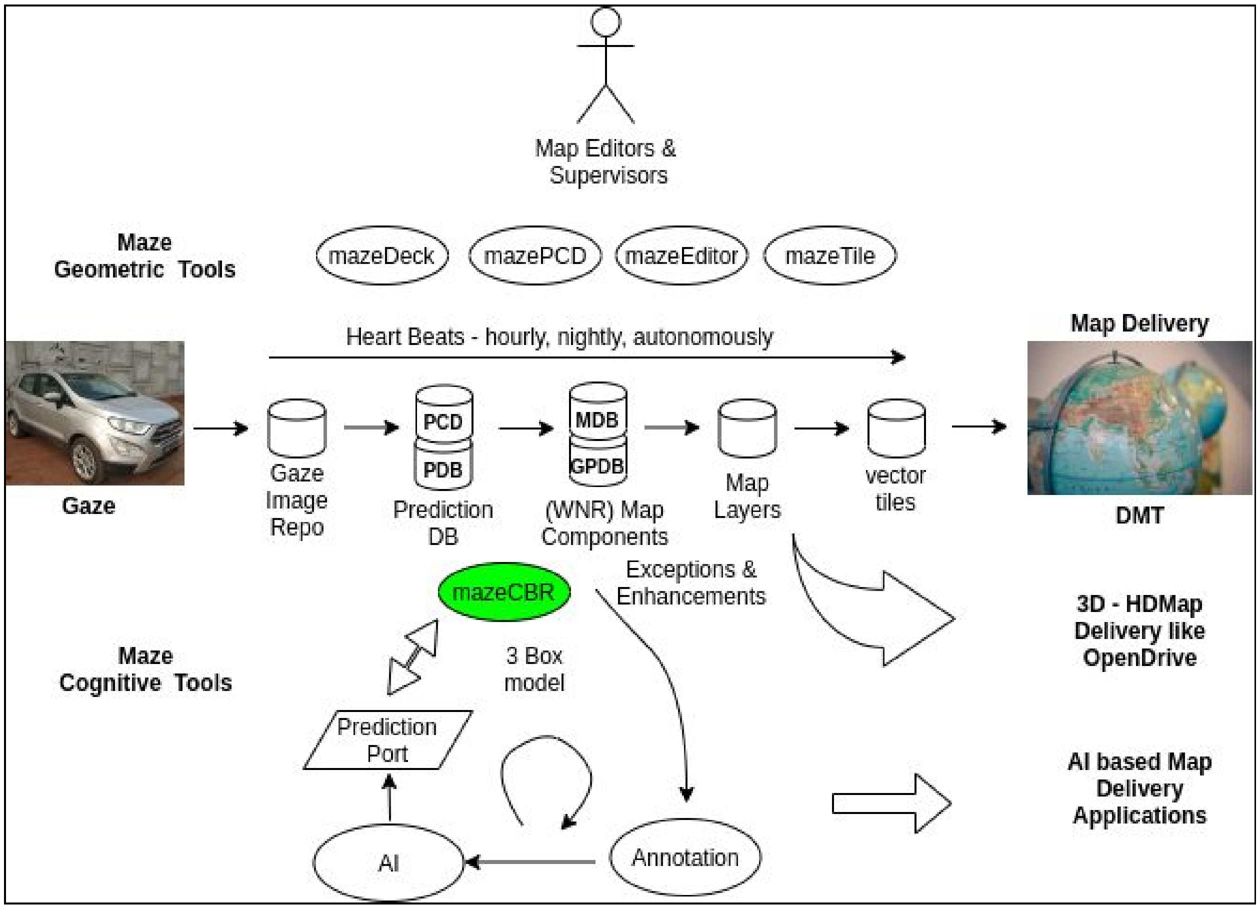


Figure 1: MapmyIndia - 3 Box Model

3 Theory and Algorithms

CBIR, a computer vision technique for image retrieval, is developed and implemented to improve the overall performance of the maps. An existing optimized version of Mask R-CNN is being implemented by the company. The same will be used for feature extraction and CBIR implementation. A Feature Database will be designed and scripted to store the feature extracted from the Mask R-CNN running on python. A CBIR model will be implemented on python utilizing various necessary modules of deep learning to retrieve similar images. Evaluation metrics like mAP is used extensively to evaluate the performance of the overall model.

3.1 Computer Vision

Computer Vision is the most actively growing field currently in computer science. It focuses on replicating parts of the complexity of the human vision system and enabling computers to identify and process objects in images and videos in the same way that humans do. A multidisciplinary field broadly classified as a subfield of Artificial Intelligence and Machine Learning, computer vision involves use of specialized methods and general learning algorithms.

3.1.1 Manual Processing

Before the advent of deep learning, the tasks that computer vision could perform were very limited and required a lot of manual coding and effort by developers and human operators. For instance, if you wanted to perform facial recognition, you would have to perform the following steps:

- **Create a database**: You had to capture individual images of all the subjects you wanted to track in a specific format.
- **Annotate images**: Then for every individual image, you would have to enter several key data points, such as distance between the eyes, the width of nose bridge, distance between upper-lip and nose, and dozens of other measurements that define the unique characteristics of each person.

- **Capture new images**: Next, you would have to capture new images, whether from photographs or video content. And then you had to go through the measurement process again, marking the key points on the image. You also had to factor in the angle the image was taken.

3.1.2 Deep Learning

Machine learning provided a different approach to solving computer vision problems. With machine learning, developers no longer needed to manually code every single rule into their vision applications. Instead they programmed "features," smaller applications that could detect specific patterns in images. They then used a statistical learning algorithm such as linear regression, logistic regression, decision trees or support vector machines (SVM) to detect patterns and classify images and detect objects in them.

Deep learning provided a fundamentally different approach to doing machine learning. Deep learning relies on neural networks, and provided with many labeled examples of a specific kind of data, it'll be able to extract common patterns between those examples and transform it into a mathematical equation that will help classify future pieces of information.

Most current computer vision applications such as cancer detection, self-driving cars and facial recognition make use of deep learning.

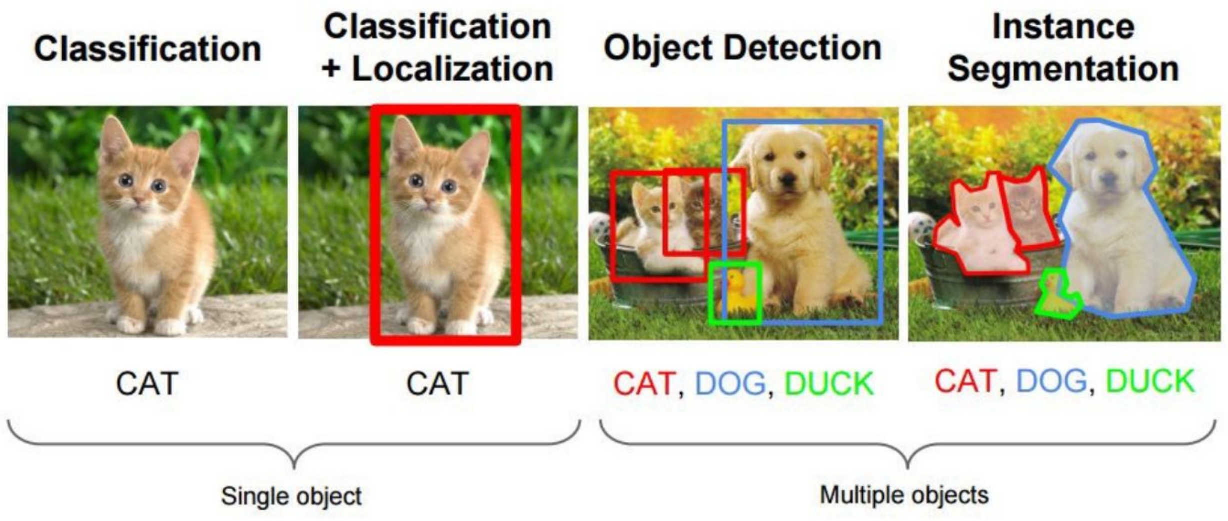


Figure 2: Computer Vision Applications.

3.2 Mask R-CNN

3.2.1 R-CNN

Traditional methods involved selecting a huge no. of regions in an image which lead to inefficiency in terms of time and cost. A new method was proposed where we use selective search to extract just 2000 regions from the image, called region proposals.

These 2000 candidate region proposals are warped into a square and fed into a convolutional neural network that produces a 4096-dimensional feature vector as output. The CNN works as a feature extractor and the dense output layer consists of these features extracted from the image which are then fed into an SVM to classify the presence of the object within that candidate region proposal. In addition to object detection within the region proposals, the algorithm also predicts four values which are offset values to increase the precision of the bounding box.

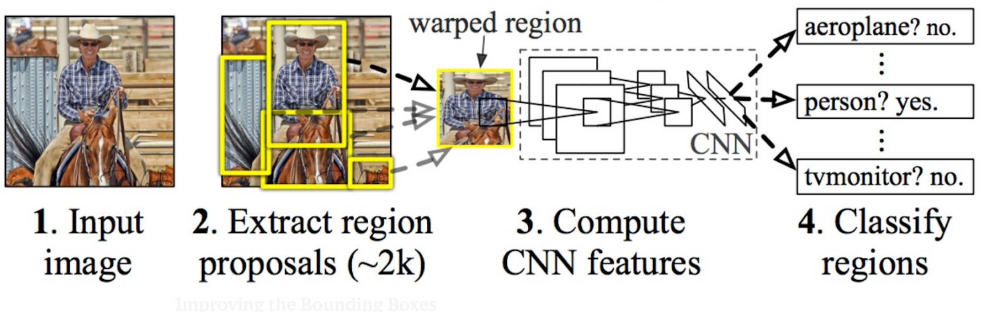


Figure 3: R-CNN Architecture.

Problems with R-CNN:-

- It still takes a huge amount of time to train the network as you would have to classify 2000 region proposals per image.
- It cannot be implemented real time as it takes around 47 seconds for each test image.
- The selective search algorithm is a fixed algorithm. Therefore, no learning is

happening at that stage. This could lead to the generation of bad candidate region proposals.

3.2.2 Fast R-CNN

The approach to Fast R-CNN algorithm is similar to the R-CNN algorithm, but instead of region proposals being fed into CNN, the input image is feeded to the CNN to generate a convolutional feature map. From the convolutional feature map , the region proposals are identified and are then warped into squares and by using the RoI pooling layer, they are reshaped into a fixed size that 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.

Fast R-CNN is faster than R-CNN because 2000 region proposals don't have to be fed into the CNN every time. Instead, the convolutional operation is done only once per image and a feature map is generated from it.

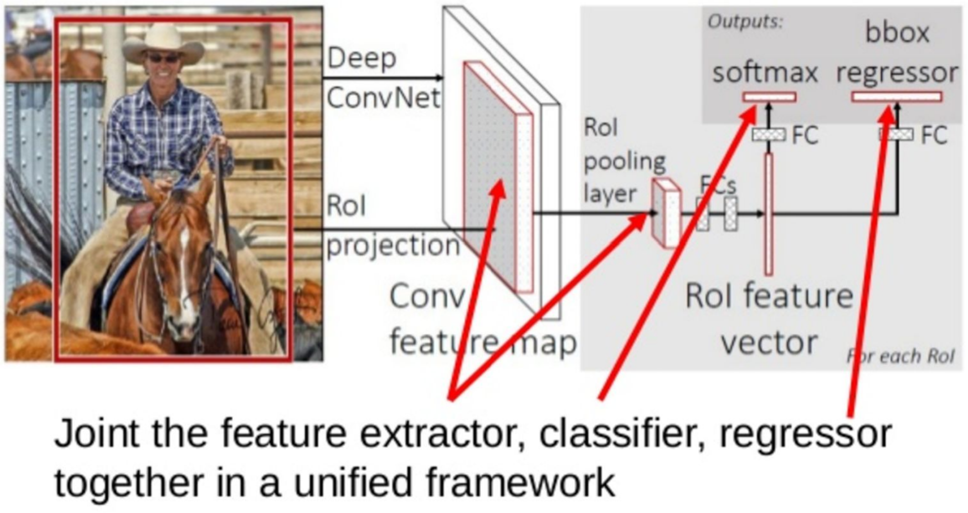


Figure 4: Fast R-CNN Architecture.

3.2.3 Faster R-CNN

Both R-CNN and Fast R-CNN use selective search to find out the region proposals. Selective search is a slow and time-consuming process affecting the performance of the network. Faster R-CNN is an object detection algorithm that eliminates the selective search algorithm and lets the network learn from region proposals.

Similar to Fast R-CNN, the image is fed as an input to a convolutional network which provides the convolutional feature maps. Instead of identifying region proposals on the feature map using selective search algorithms, a separate network, Region Proposal Network, is used to predict the region proposals. Those 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.

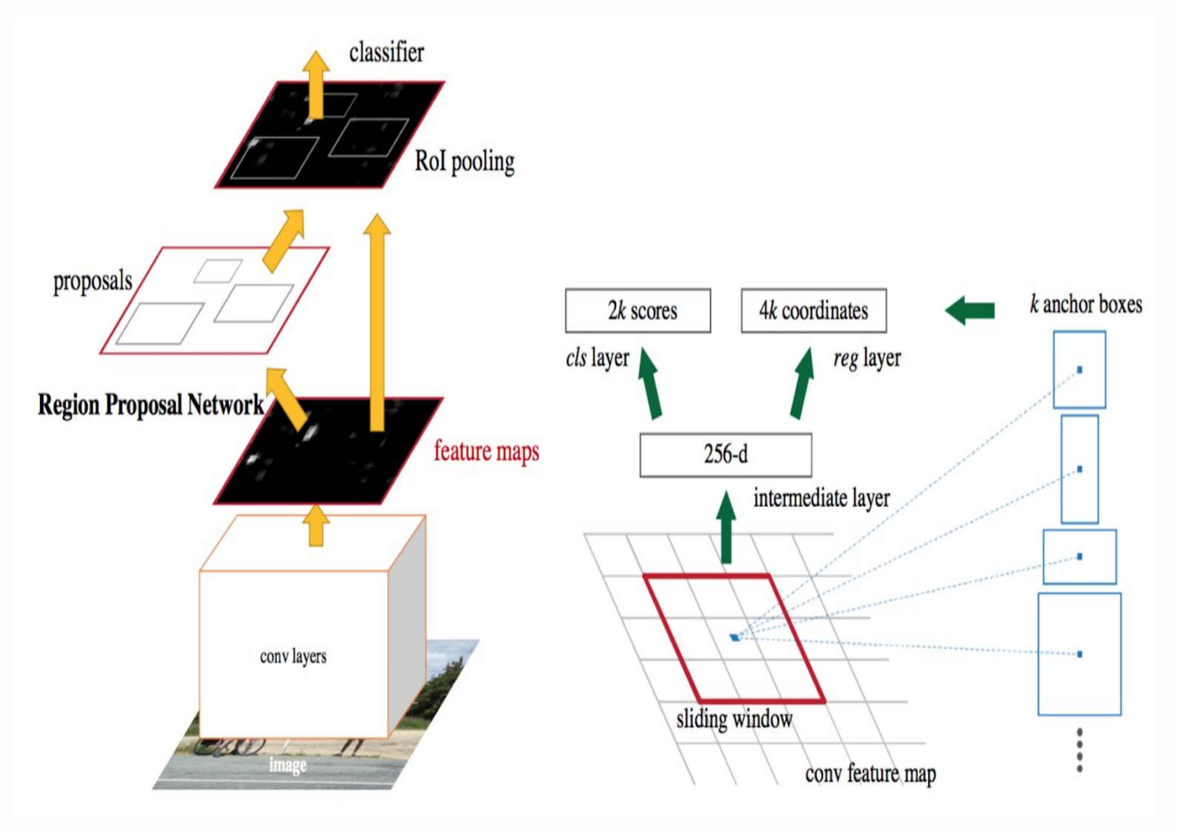


Figure 5: Faster R-CNN Architecture

3.2.4 Mask R-CNN

Mask R-CNN is an extension of Faster R-CNN and its framework is built on top of it. It is aimed at solving instance segmentation problems as the other CNN algorithms can only detect the objects but not segment them for each instance with a mask. In other words, it can separate different objects in an image or a video. You give it an image, it gives you the object bounding boxes, classes and masks. Mask R-CNN has an improved performance time and reduces the need of a different model for segmentation of the image. It was modelled on the COCO dataset but is extensively being used by MapmyIndia with the optimizations aligning with their utility.

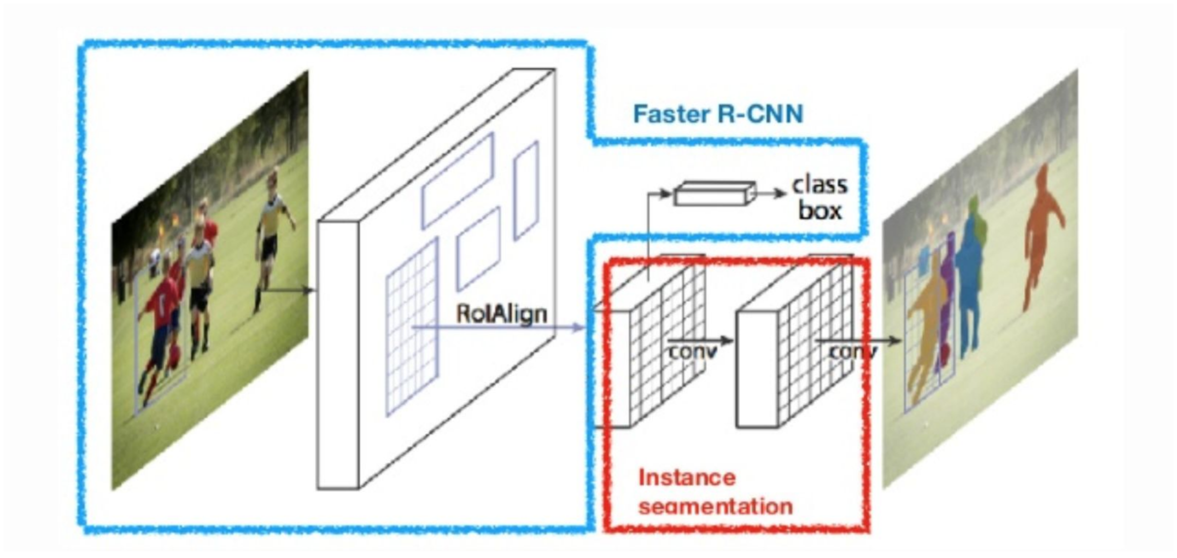


Figure 6: Mask R-CNN Architecture - Faster R-CNN with image segmentation

Another major contribution of Mask R-CNN is the refinement of the ROI pooling. In ROI, the warping is digitalized: the cell boundaries of the target feature map are forced to realign with the boundary of the input feature maps. Therefore, each target cell may not be in the same size. Mask R-CNN uses ROI Align which does not digitalize the boundary of the cells and make every target cell to have the same size (bottom right). It also applies interpolation to calculate the feature map values within the cell better.

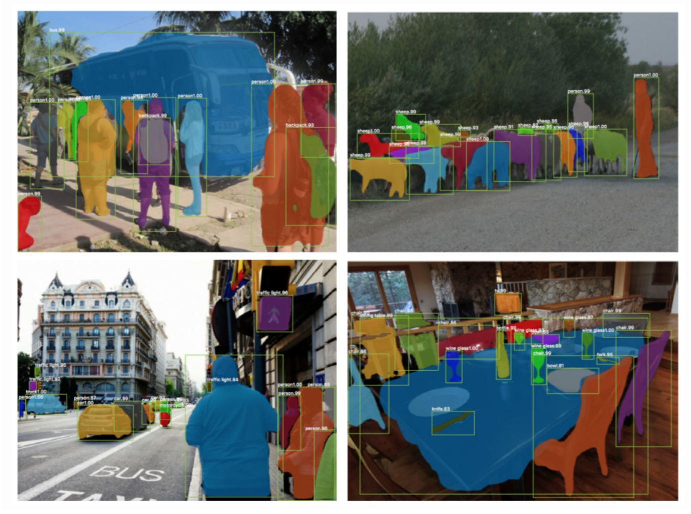


Figure 7: Predictions by Mask R-CNN - Object Detection and added bounding boxes.

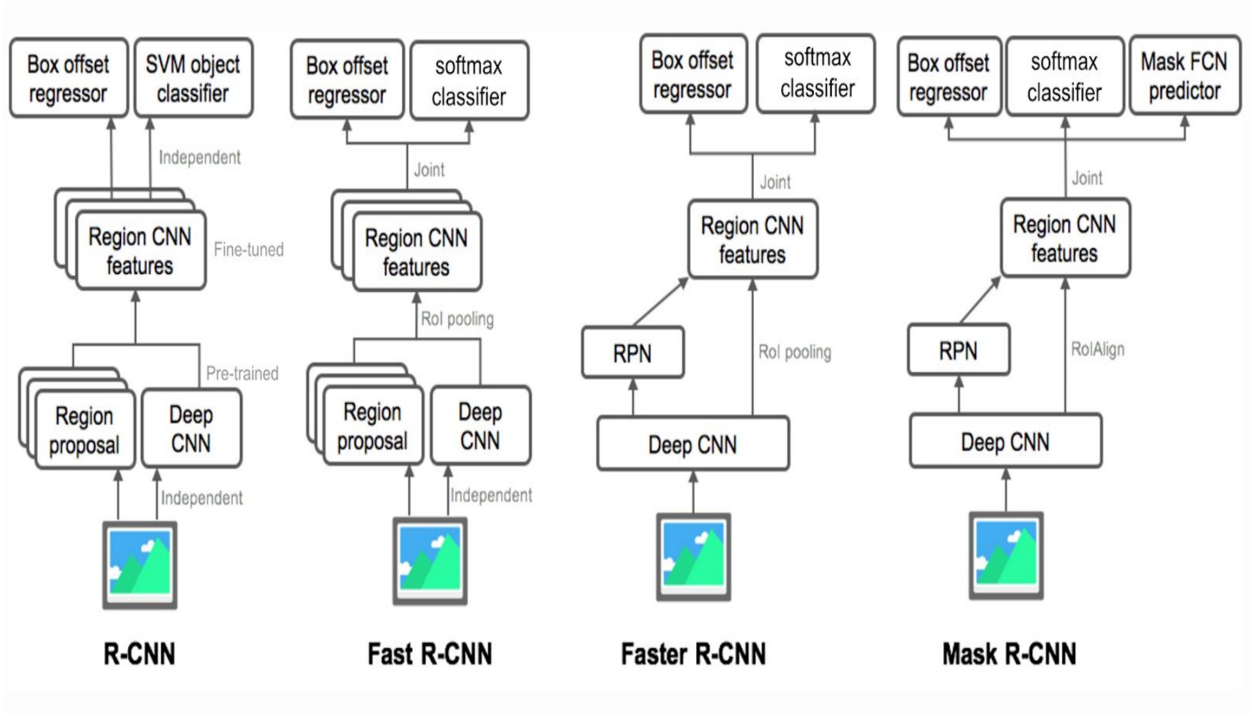


Figure 8: Summary of Models and their Architectures.

3.3 DATABASE

3.3.1 MongoDB

MongoDB is a cross-platform document-oriented database program. It is classified as NoSQL database program as it uses JSON-like documents (BSON) with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under Server Side Public License (SSPL). It is a widely accepted database structure in various industries, with even banks adapting it. It also provides cloud-based, enterprise-based services making it easily accessible. We use the local version of it to create a database that stores the image feature vectors in .BSON format. A pymongo extension is used to run the script and queries in python.

3.4 CBIR

3.4.1 Introduction

Content Based Image Retrieval, also known as Query by Image Content (QBIC), presents the technologies allowing to organize digital pictures by their visual features. They are based on the application of computer vision techniques to the image retrieval problem in large databases. Content-Based Image Retrieval (CBIR) consists of retrieving the most visually similar images to a given query image from a database of images. "Content-based" means that the search analyzes the contents of the image rather than the metadata such as keywords, tags, or descriptions associated with the image. The term "content" in this context might refer to colors, shapes, textures, or any other information that can be derived from the image itself. CBIR is desirable because searches that rely purely on metadata are dependent on annotation quality and completeness.

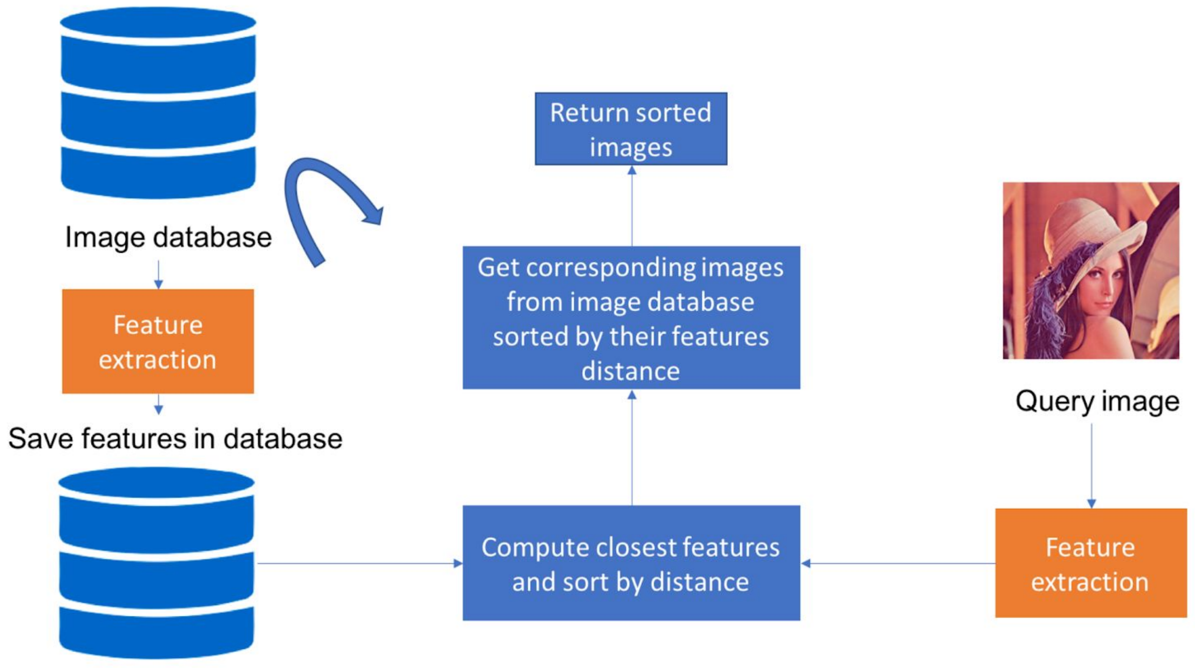


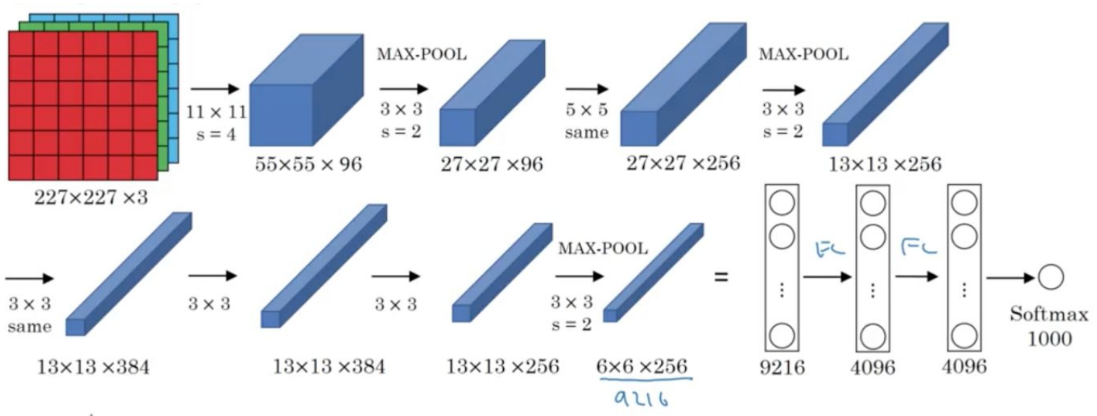
Image database
Feature extraction
Save features in database
Return sorted images
Get corresponding images from image database sorted by their features distance
Compute closest features and sort by distance
Query image
Feature extraction

Figure 10: Content Based Image Retrieval Architecture

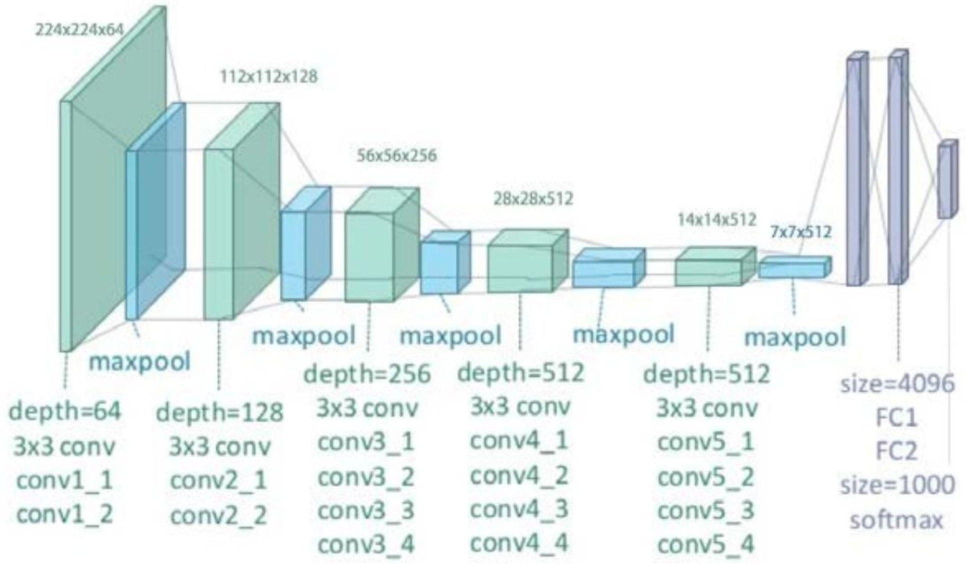
3.4.2 PyTorch

PyTorch is an open-source machine learning library used for developing and training neural network based deep learning models. Primarily developed by Facebook's AI research group, it can be used with Python as well as C++, with Python being more polished. Pytorch (backed by biggies like Facebook, Microsoft, SalesForce, Uber) is immensely popular in research labs. Not yet on many production servers that are ruled by frameworks like TensorFlow (Backed by Google), Pytorch is picking up fast. Unlike most other popular deep learning frameworks like TensorFlow, which use static computation graphs, PyTorch uses dynamic computation, which allows greater flexibility in building complex architectures. Pytorch uses core Python concepts like classes, structures and conditional loops — that are a lot familiar to our eyes, hence a lot more intuitive to understand. This makes it a lot simpler than other frameworks like TensorFlow that bring in their own programming style.

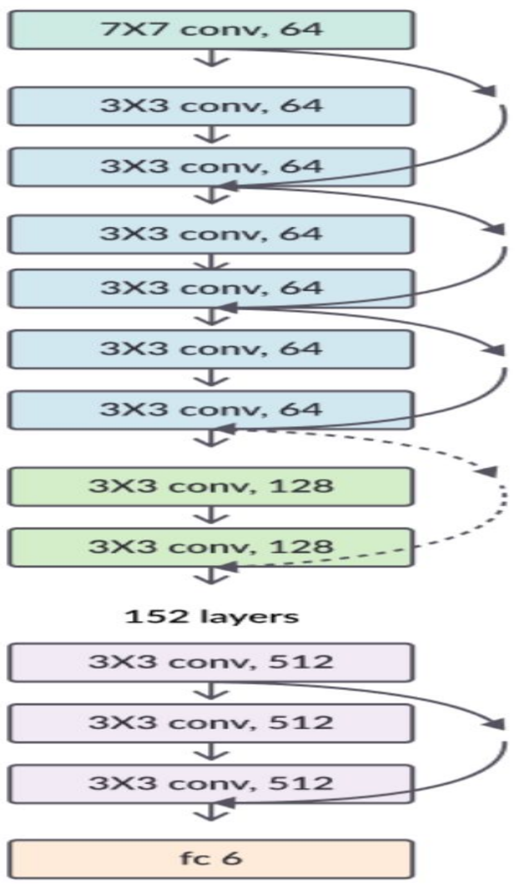
PyTorch on python was used as the base framework. AlexNet, VGG19, ResNet152 models pre-trained on ImageNet dataset were used. A new image dataset, IndoorCVPR_09, was passed through the models to extract feature vectors of the images which were stored in a Pandas Dataframe along with the image locations to be further processed on. A MapmyIndia dataset of 10,000 images comprising annotations, ground truth values, and precision, is further fine-tuned on the pretrained PyTorch models to extract image feature vectors.



a) AlexNet



b) VGG19



c) ResNet

Figure 11: PyTorch Neural Network Architectures

3.4.3 Annoy Algorithm

Annoy (Approximate Nearest Neighbour Oh Yeah), developed by Spotify, is C++ library with Python bindings to search for points in space that are close to a given query point. It also creates large read-only file-based data structures that are mmapped into memory so that many processes may share the same data. Annoy is almost as fast as the fastest libraries with the ability to use static files as indexes. It works by using random projections and by building up a tree. At every intermediate node in the tree, a random hyperplane is chosen, which divides the space into two subspaces. This hyperplane is chosen by sampling two points from the subset and taking the hyperplane equidistant from them. We do this k times so that we get a forest of trees. k has to be tuned to our needs. Various distance and similarity measures like Euclidean Distance, Cosine Similarity, Hamming Distance, Dot Product are available to perform the search.

The pre-trained PyTorch models provide image feature vectors which are given as the input to the Annoy Library. We use k=20 to develop a forest of 20 trees, with Euclidean distance as the metric for similar images, and giving 1000 as the dimensions. It results in indexes of the images which are similar to our query image after 196 minutes of processing time on the CPU.

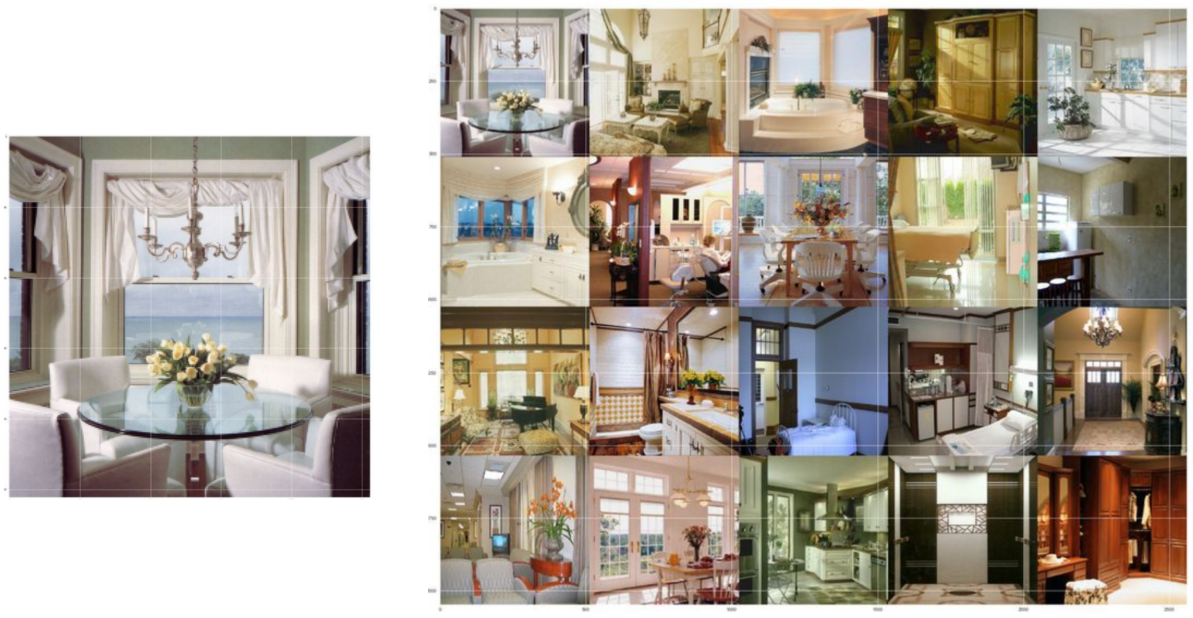


Figure 12: Image Retrieval similar to query image using Annoy Algorithm.

3.4.4 KNN

K-nearest neighbour (KNN) algorithm is a supervised machine learning algorithm that can be used to solve both classification and regression problems. The KNN algorithm assumes that similar images exist in close proximity. *k*-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, normalizing the training data can improve its accuracy dramatically. A peculiarity of the *k*-NN algorithm is that it is sensitive to the local structure of the data.

We use the extracted image feature vectors from fine-tuned on the pre-trained models of PyTorch for the input data. A query image feature vector is passed as the input query, to which distance, calculated using the Euclidean distance equation, of each image vector in the dataset is calculated and stored in a list. The resulting list is sorted and the initial few values, in this case 10, is used to extract the feature vectors close to the query image. The image indexes are used to further build a montage of the similar images.

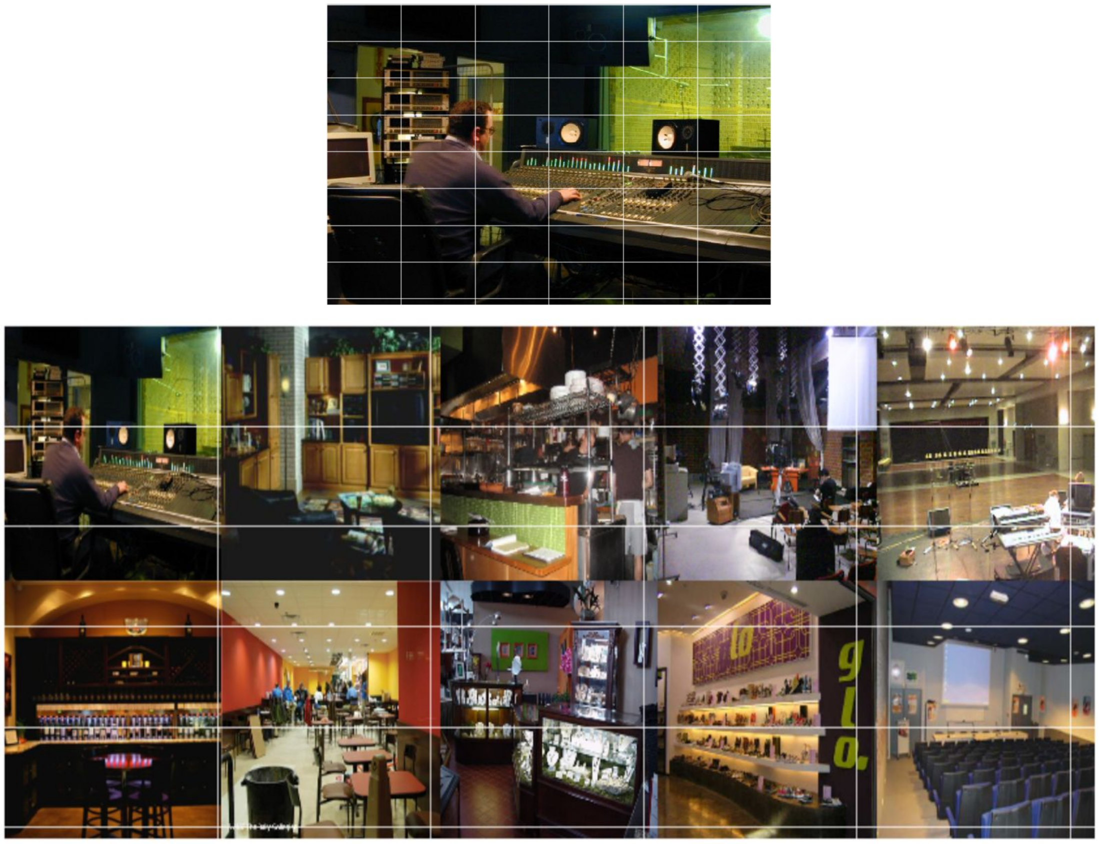


Figure 13: Image Retrieval similar to query image using KNN Algorithm.

3.5 Implementation Protocol

GitHub Link: https://github.com/avtanshpandey/CBIR_Deep-Learning

- **Step1: Extracting Feature Vectors**

  The pytorch models are tapped at the layer penultimate layer to extract and store features of the images and perform CBIR.

- **Step 2: Designing and Populating Feature Database**

  A Feature Database designed on MongoDB is used to store the extracted features of the images. It will be populated with the features of Gaze Images after running the predictions.

- **Step 3: Annotated Images**

  We use the seeding dataset that is annotated and has predictions. We use it to identify the images resulting in False Positives and push it as queries into the CBIR model.

- **Step 4: CBIR Model**

  The images received are used as queries and similar images are extracted from the feature database (FDB) using distance metrics such as Euclidean or Cosine on algorithms like Annoy, KNN, and are then sent back to the annotation team to re-annotate them and then calculate the improvement in the mAP.

- **Step 5: mAP Calculation**

  mAP for the superset of the dataset passed through the Mask R-CNN model is compared to the subset of it, i.e., the whole dataset minus the query image dataset, and also another subset, i.e., the query image dataset. The comparison is studied to evaluate the effectiveness of the CBIR implementation for improvement in the accuracy of the 3-Box model.

3.6 Evaluation Metrics

Various evaluation methods will be used to determine the functioning of our model and to calculate the improvement due to CBIR.

Following are the few evaluative metrics being:-

- **Precision**

  It answers what proportion of positive identifications was actually correct.

  Precision = $\frac{True\ Positives}{True\ Positives + False\ Positives}$

- **Recall**

  It answers, what proportion of actual positives was identified correctly.

  Recall = $\frac{True\ Positives}{True\ Positives + False\ Negatives}$

- **F1-Score**

  It is a measure of the test's accuracy and conveys the balance between precision and recall of the test.

  F1-Score = $2 \times \frac{Precision \times Recall}{Precision + Recall}$

- **IoU**

  Intersection over Union is an evaluation metric used to measure the accuracy of an object detector on a particular dataset.

- **mAP**

  AP (Average precision) is a popular metric in measuring the accuracy of object detectors like Faster R-CNN, SSD, etc. Average precision computes the average precision value for recall value over 0 to 1. mAP (mean average precision) is the average of AP. AP is averaged over all categories. Traditionally, this is called "mean average precision" (mAP). We make no distinction between AP and mAP (and likewise AR and mAR) and assume the difference is clear from context.

4 CONCLUSION

This project is aimed at designing, developing and implementing a Content Based Image Retrieval model for improving the performance of the Map Generation architecture. The project started with understanding and using the MapmyIndia's API and SDK functions and features for which an application was developed around it using React-Native techstack.

Project being on Computer Vision required acquiring knowledge of it. The existing model uses an optimised version of Mask R-CNN, so a deeper understanding of the technique and algorithm was acquired which will be further used for acquiring image features. CBIR being the key concept of the project was researched upon and existing technologies were used as an inspiration to understand the possible implementations.

A complete architecture has been devised to go about the implementation of the CBIR model which involves designing an additional database to store the features and use it to extract the images based on the optimal distance metrics learnt.

Image retrieval model successfully performed on an open IndoorCVPR_09 dataset and on the MapmyIndia gaze image dataset. Backed by various research papers, this model is aimed to improve the overall performance by 10%. This can be calculated using mAP evaluation on the dataset using ground truth and prediction values of the gaze images.

References

1. Mask R-CNN on ShortScience.org. Retrieved September 3, 2020, from https://www.shortscience.org/paper?bibtexKey=journals/corr/HeGDG17
2. Mask R-CNN on ShortScience.org. Retrieved September 3, 2020, from https://www.shortscience.org/paper?bibtexKey=journals/corr/HeGDG17
3. Tesla's Data Engine and what we should learn from it. Retrieved September 4, 2020, from https://www.braincreators.com/brainpower/insights/teslas-data-engine-and-what-we-should-all-learn-from-it
4. (2020, March 23). Tesla files patent for sourcing self-driving training ... - Electrek. Retrieved September 4, 2020, from https://electrek.co/2020/03/23/tesla-patent-sourcing-self-driving-training-data/
5. Tesla's Deep Learning at Scale: Using Billions of Miles to .... Retrieved September 5, 2020, from https://towardsdatascience.com/teslas-deep-learning-at-scale-7eed85b235d3
6. Content-based image retrieval using color and texture fused .... Retrieved September 15, 2020, from https://www.sciencedirect.com/science/article/pii/S0895717710005352
7. (2017, September 14). Keras Tutorial: Content Based Image Retrieval ... - Medium. Retrieved September 20, 2020, from https://medium.com/sicara/keras-tutorial-content-based-image-retrieval-convolutional-denoising-autoencoder-dc91450cc511
8. Scale-invariant feature transform - Wikipedia. Retrieved September 23, 2020, from https://en.wikipedia.org/wiki/Scale-invariant_feature_transform
9. Using Very Deep Autoencoders for Content-Based Image .... Retrieved October 6, 2020, from http://www.cs.toronto.edu/~fritz/absps/esann-deep-final.pdf
10. (2017, February 27). (PDF) Detection of Nuclei in H&E Stained Sections Using .... Retrieved October 8, 2020, from https://www.researchgate.net/publication/314090304_Detection_of_Nuclei_in_HE_Stained_Sections_Using_Convolutional_Neural_Networks

11. 8290434.pdf - CS230 - Stanford University. Retrieved October 15, 2020, from https://cs230.stanford.edu/projects_spring_2018/reports/8290434.pdf
12. https://books.google.co.in/books?hl=en&lr=&id=Yo1xDwAAQBAJ&oi=fnd&pg=PA71&dq=mask+r-cnn+for+cbir&ots=rIqlpHXwoF&sig=zctkh5PtQ3RczrFvoNwbf3VW_iA#v=onepage&q=mask%20r-cnn%20for%20cbir&f=false
13. Everything You Ever Wanted To Know About Computer Vision .... Retrieved October 28, 2020, from https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e
14. Simple Approximate Nearest Neighbors in Python with Annoy .... Retrieved October 30, 2020, from https://medium.com/@kevin_yang/simple-approximate-nearest-neighbors-in-python-with-annoy-and-lmdb-e8a701baf905
15. Approximate Nearest Neighbors Methods for ... - Research. Retrieved November 2, 2020, from http://groups.csail.mit.edu/vision/vip/nips03ann/abstracts/
16. (2017, September 2). Recent Advance in Content-based Image Retrieval: A ... - arXiv. Retrieved November 4, 2020, from https://arxiv.org/pdf/1706.06064
17. Fast Approximate Nearest-Neighbor Search with k ... - IJCAI. Retrieved November 4, 2020, from https://www.ijcai.org/Proceedings/11/Papers/222.pdf
18. GitHub - spotify/annoy: Approximate .... Retrieved November 12, 2020, from https://github.com/spotify/annoy
19. pytorch/pytorch: Tensors and Dynamic .... Retrieved November 12, 2020, from https://github.com/pytorch/pytorch
20. (2018, March 26). K Nearest Neighbor | KNN Algorithm .... Retrieved November 19, 2020, from https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/
21. AlexNet: The Architecture that Challenged CNNs | by Jerry .... Retrieved November 23, 2020, from https://towardsdatascience.com/alexnet-the-architecture-that-challenged-cnns-e406d5297951

22. ImageNet Classification with Deep Convolutional Neural .... Retrieved November 24, 2020, from https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf
23. An Overview of ResNet and its Variants - Towards Data Science. Retrieved November 24, 2020, from https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035
24. Deep Residual Learning for Image .... Retrieved November 26, 2020, from https://arxiv.org/abs/1512.03385
25. Understanding the VGG19 Architecture - OpenGenus IQ. Retrieved November 28, 2020, from https://iq.opengenus.org/vgg19-architecture/
26. (2020, May 2). PyRetri: A PyTorch-based Library for Unsupervised Image .... Retrieved November 28, 2020, from https://arxiv.org/abs/2005.02154