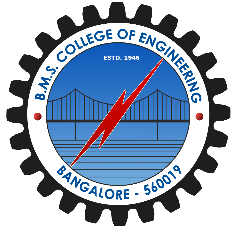
**B.M.S. College of Engineering**

***(Autonomous Institution affiliated to VTU, Belagavi)***

**BMSCE ACM Student Chapter in collaboration with TechGig**

**Hackathon**

**(Hack for Good)**

**Team: Byte Masters**

Likhith G S

M Ananta Naga Rajesh

Krishn Maloo

Kshitij S

**Theme: Public Safety and Crime Prevention**

**Topic: Crime Hotspot Detection using location data and Image Processing**

**Introduction:**

Public safety plays a critical role in crime prevention and maintaining a peaceful and secure society. It encompasses a range of measures, policies, and practices aimed at safeguarding individuals and communities from harm, including criminal activities.

Crime in India is a complex issue that arises from a combination of social, economic, cultural, and governance factors. India, being a diverse and populous country with varying socio-economic conditions, experiences a wide range of criminal activities. Addressing crime in India requires a comprehensive approach that includes legal reforms, investment in law enforcement infrastructure and training, addressing socio-economic disparities, promoting education and awareness, and fostering community engagement. While progress has been made in certain areas, there is ongoing work needed to create safer communities and ensure justice for all.

**Problem Statement:**

Crime prevention is a critical aspect of maintaining public safety and security. Law enforcement agencies and communities strive to allocate resources efficiently to combat crime. Understanding patterns and trends in crime occurrence can significantly aid in proactive policing efforts.

**Solution:**

In our current project, our team of four individuals immersed in the realm of computer vision, with a specific emphasis on the intricate field of object detection, particularly centred around the identification of pistols within images. Our approach involves harnessing the power of the advanced YOLOv5 algorithm, meticulously implemented through the PyTorch deep learning framework. The YOLOv5 algorithm, renowned for its agility, excels in rapidly and accurately identifying objects by delineating bounding boxes around them and ascribing pertinent class labels.

Our strategic utilization of PyTorch, a flexible and widely embraced deep learning platform, empowers us to seamlessly construct and fine-tune the YOLOv5 model, thereby augmenting its proficiency in detecting pistols. The implications of this project are profound, spanning across security applications, law enforcement, and safety protocols, as the efficacy of meticulous object detection technology holds pivotal significance in real-world scenarios.

KMeans is a clustering algorithm used for grouping similar data points into a specified number of clusters. It works by iteratively assigning data points to the nearest cluster centre and updating the centres based on the mean of the assigned points. The goal is to minimize the sum of squared distances between data points and their assigned cluster centres.

Using a KMeans model to detect crime hotspots using location data is a common approach in data analysis and crime prediction. KMeans is an unsupervised machine learning algorithm that aims to partition data points into clusters based on their similarity. In the context of crime hotspot detection, it can help identify areas with similar crime patterns, which can be interpreted as hotspots.

**Work Flow K-means model**

Data Collection:

* We gathered a datasets containing information related to all the types of crime in those states in different years and combined them.

Data Preprocessing:

* Clean and Transform Data: Remove missing values, outliers, and irrelevant entries. Format the data for effective clustering.
* Scale and Select Features: Normalize features to comparable scales. Choose relevant attributes like crime type, time, and location.
* Optimal Cluster Number (K): Determine the suitable number of clusters using methods like Elbow or Silhouette Score to ensure meaningful crime pattern identification.

Model Selection:

* We chose KMeans Clustering Machine learning model to detect the high and low crime regions.

Model Training:

* Center Initialization: Begin by placing cluster centers, often using random or KMeans initialization.
* Iterative Refinement: Alternate between assigning data points to nearest centers and updating centers based on assigned points' means.
* Convergence Check: Halt training when assignments and centers stabilize, typically determined by a maximum iteration limit or minimal changes.

Evaluation and Post-Processing:

* Cluster Interpretation: Analyze clusters to extract insights about crime patterns and characteristics.
* Visualize and Apply: Use visualizations to understand cluster distributions and translate findings into actionable strategies for crime prevention and resource allocation.

**Code:**

import numpy as np # linear algebra

import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)

import matplotlib.pyplot as plt # for data visualization

import seaborn as sns # for statistical data visualization

%matplotlib inline

import warnings

warnings.filterwarnings('ignore')

data1 = pd.read\_csv('/content/01\_District\_wise\_crimes\_committed\_IPC\_2001\_2012.csv')

data2 = pd.read\_csv('/content/01\_District\_wise\_crimes\_committed\_IPC\_2013.csv')

data = pd.concat([data1,data2])

data.dropna(subset=['STATE/UT'], inplace=True)

data=data.dropna(axis = 1)

data['STATE/UT'] = data['STATE/UT'].astype('|S80')

data['DISTRICT']=data['DISTRICT'].astype('|S80')

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

data['STATE/UT'] = le.fit\_transform(data['STATE/UT'])

data['DISTRICT'] = le.fit\_transform(data['DISTRICT'])

df1 = data[['STATE/UT', 'MURDER']]

df2 = data[['STATE/UT', 'RAPE']]

df3 = data[['STATE/UT', 'KIDNAPPING & ABDUCTION']]

df4 = data[['STATE/UT', 'DACOITY']]

df5 = data[['STATE/UT', 'ROBBERY']]

df6 = data[['STATE/UT', 'RIOTS']]

df7 = data[['STATE/UT', 'DOWRY DEATHS']]

df1['MURDER'] = df1.groupby('STATE/UT')['MURDER'].transform('sum')

df2['RAPE'] = df2.groupby('STATE/UT')['RAPE'].transform('sum')

df3['KDINAPPING & ABDUCTION'] = df3.groupby('STATE/UT')['KIDNAPPING & ABDUCTION'].transform('sum')

df4['DACOITY'] = df4.groupby('STATE/UT')['DACOITY'].transform('sum')

df5['ROBBERY'] = df5.groupby('STATE/UT')['ROBBERY'].transform('sum')

df6['RIOTS'] = df6.groupby('STATE/UT')['RIOTS'].transform('sum')

df7['DOWRY DEATHS'] = df7.groupby('STATE/UT')['DOWRY DEATHS'].transform('sum')

df1 = df1.drop\_duplicates(subset=['STATE/UT'])

df2 = df2.drop\_duplicates(subset=['STATE/UT'])

df3 = df3.drop\_duplicates(subset=['STATE/UT'])

df4 = df4.drop\_duplicates(subset=['STATE/UT'])

df5 = df5.drop\_duplicates(subset=['STATE/UT'])

df6 = df6.drop\_duplicates(subset=['STATE/UT'])

df7 = df7.drop\_duplicates(subset=['STATE/UT'])

X = df1

from sklearn.cluster import KMeans

kmeans = KMeans(n\_clusters=2,random\_state=0)

kmeans.fit(X)

labels = kmeans.labels\_

clusters = kmeans.cluster\_centers\_

y\_km = kmeans.fit\_predict(df1)

df11 = df1

df11 = np.array(df11)

plt.scatter(df11[y\_km == 0,0],df11[y\_km == 0,1], s = 50, color = 'red')

plt.scatter(df11[y\_km == 1,0],df11[y\_km == 1,1], s = 50, color = 'blue')

plt.scatter(clusters[0][0], clusters[0][1],marker = '\*',s=200, color ='black')

plt.scatter(clusters[1][0], clusters[1][1],marker = '\*',s=200, color ='black')

plt.title("Murders")

plt.xlabel("State labels")

plt.ylabel("Number of Murders")Work Flow Image processing model

X = df2

from sklearn.cluster import KMeans

kmeans = KMeans(n\_clusters=2,random\_state=0)

kmeans.fit(X)

labels = kmeans.labels\_

clusters = kmeans.cluster\_centers\_

y\_km = kmeans.fit\_predict(df2)

df11 = df2

df11 = np.array(df11)

plt.scatter(df11[y\_km == 0,0],df11[y\_km == 0,1], s = 50, color = 'red')

plt.scatter(df11[y\_km == 1,0],df11[y\_km == 1,1], s = 50, color = 'blue')

plt.scatter(clusters[0][0], clusters[0][1],marker = '\*',s=200, color ='black')

plt.scatter(clusters[1][0], clusters[1][1],marker = '\*',s=200, color ='black')

plt.title("Rape")

plt.xlabel("State labels")

plt.ylabel("Number of Rapes")

clusters

**Work Flow of Image Processing Model**

Data Collection:

* We gathered a dataset of images that contain instances of pistols.

Data Annotation:

* Annotated the collected images using tools provided by Roboflow platforms. Draw bounding boxes around the pistols in the images to create ground truth labels for training.

Data Preprocessing:

* Resized and normalized the images to ensure consistency in size and pixel values.
* Augmented the dataset by applying transformations like rotation, flipping, and brightness adjustments. This increased the diversity of the data and helps improve model generalization.

Model Selection:

* We chose YOLO v5 algorithm (You Only Look Once)- PyTorch as an object detection model architecture.

Model Training:

* We uploaded the annotated and preprocessed dataset to the Roboflow platform.
* Configured the training settings, including the chosen model architecture, batch size, learning rate, and number of epochs.
* Initiated the training process. The model learns to detect pistols based on the annotated data

Evaluation and Post-Processing:

* After training, evaluate the model's performance using evaluation metrics like precision, recall, and mean Average Precision (mAP).
* Apply non-maximum suppression (NMS) to post-process the model's predictions and eliminate duplicate or highly overlapping bounding boxes.

**Code:**

# clone YOLOv5 repository

!git clone https://github.com/ultralytics/yolov5 # clone repo

%cd yolov5

!git reset --hard 064365d8683fd002e9ad789c1e91fa3d021b44f0

# install dependencies as necessary

!pip install -qr requirements.txt # install dependencies (ignore errors)

import torch

from IPython.display import Image, clear\_output # to display images

from utils.downloads import attempt\_download # to download models/datasets

# clear\_output()

print('Setup complete. Using torch %s %s' % (torch.\_\_version\_\_, torch.cuda.get\_device\_properties(0) if torch.cuda.is\_available() else 'CPU'))

!pip install roboflow

from roboflow import Roboflow

rf = Roboflow(api\_key="CIXwFXaijriiKmeR5dJn")

project = rf.workspace("school-gh2wa").project("byte-masters")

dataset = project.version(1).download("yolov5")

%cd /content/yolov5

%cat {dataset.location}/data.yaml

# define number of classes based on YAML

import yaml

with open(dataset.location + "/data.yaml", 'r') as stream:

num\_classes = str(yaml.safe\_load(stream)['nc'])

%cat /content/yolov5/models/yolov5s.yaml

from IPython.core.magic import register\_line\_cell\_magic

@register\_line\_cell\_magic

def writetemplate(line, cell):

with open(line, 'w') as f:

f.write(cell.format(\*\*globals()))

%%time

%cd /content/yolov5/

!python train.py --img 416 --batch 16 --epochs 100 --data {dataset.location}/data.yaml --cfg ./models/custom\_yolov5s.yaml --weights '' --name yolov5s\_results --cache

%load\_ext tensorboard

%tensorboard --logdir runs

from utils.plots import plot\_results # plot results.txt as results.png

Image(filename='/content/yolov5/runs/train/yolov5s\_results/results.png', width=1000)

print("GROUND TRUTH AUGMENTED TRAINING DATA:")

Image(filename='/content/yolov5/runs/train/yolov5s\_results/train\_batch0.jpg', width=900)

%ls runs/

%ls runs/train/yolov5s\_results/weights

%cd /content/yolov5/

!python detect.py --weights runs/train/yolov5s\_results/weights/best.pt --img 416 --conf 0.4 --source ../test/images

import glob

from IPython.display import Image, display

for imageName in glob.glob('/content/yolov5/runs/detect/exp/\*.jpg'): #assuming JPG

display(Image(filename=imageName))

print("\n")

project.version(dataset.version).deploy(model\_type="yolov5", model\_path=f"/content/yolov5/runs/train/yolov5s\_results/")

model = project.version(dataset.version).model

#choose random test set image

import os, random

test\_set\_loc = dataset.location + "/test/images/"

random\_test\_image = random.choice(os.listdir(test\_set\_loc))

print("running inference on " + random\_test\_image)

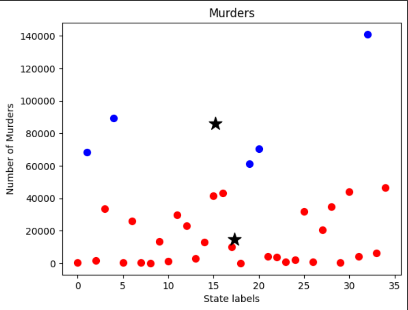
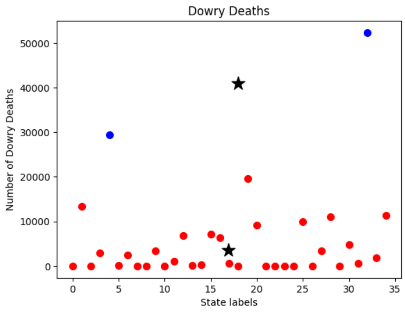
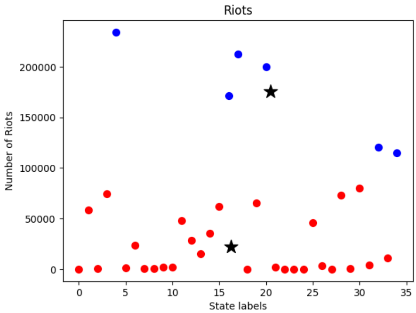
pred = model.predict(test\_set\_loc + random\_test\_image, confidence=40, overlap=30).json()

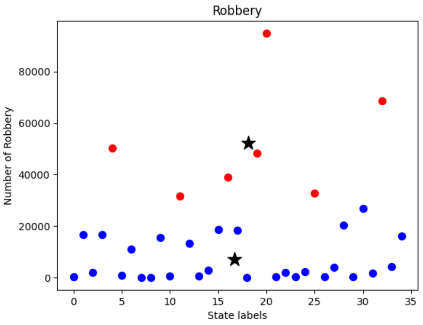
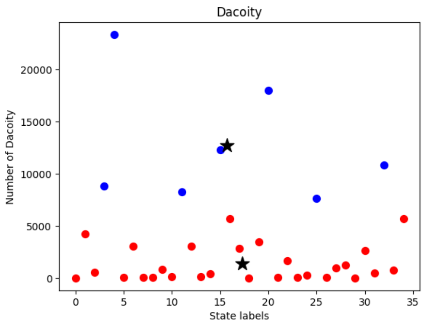
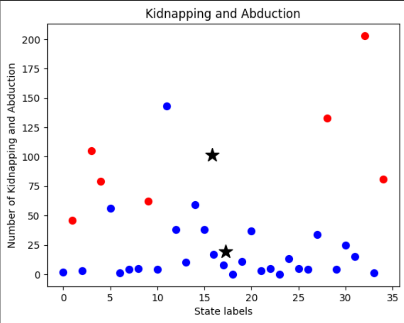
pred

**Societal impact of Kmeans model**

The use of KMeans Clustering for crime detection dividing it in 2 parts of high and low region will have following impacts:

* Resource Allocation: Efficient deployment of law enforcement resources to crime-prone areas enhances public safety while minimizing unnecessary interventions.
* Community Empowerment: Identifying crime clusters fosters community involvement, as citizens and authorities collaborate to address local safety concerns.
* Bias Awareness: The analysis of crime data underscores the importance of addressing biases in policing to ensure fair treatment across different communities.
* Policy Informed by Data: Insights from clustering support evidence-based policymaking, enabling targeted interventions and long-term crime prevention strategies.
* Privacy Protection: Handling sensitive data responsibly maintains individuals' privacy while deriving useful insights for crime mitigation.
* Social Equity: KMeans results can contribute to discussions about social inequities, potentially driving conversations and actions for broader

**Societal impact of Image Processing Model**

The use of CCTV images for detecting pistols and enhancing public safety can have several societal impacts, both positive and potentially concerning. Here are some of the key points to consider:

* Crime Deterrence: The presence of visible CCTV cameras can deter criminal activity in public spaces, as individuals may be less likely to engage in illegal or dangerous behavior if they know they are being monitored.
* Rapid Response: In the event of an incident involving a pistol, real-time monitoring of CCTV feeds can enable authorities to respond quickly, potentially preventing escalation or harm.
* Evidence Collection: CCTV images can serve as valuable evidence in criminal investigations, aiding law enforcement in identifying suspects, understanding the sequence of events, and building cases for prosecution.
* Public Safety: The use of CCTV images for pistol detection can contribute to overall public safety by identifying potentially dangerous situations and enabling timely intervention.
* Crime Prevention: By identifying pistols and suspicious behavior, law enforcement agencies can proactively address potential threats before they escalate into criminal activities.
* Surveillance in Sensitive Areas: CCTV cameras equipped with pistol detection capabilities can be strategically placed in sensitive locations such as airports, train stations, and government buildings to enhance security and protect public infrastructure.

**Conclusion:**

In conclusion, the crime hotspot detection project utilizing a KMeans model successfully identified and analyzed patterns within the provided crime data. The main objectives of the project were to determine areas with elevated crime rates and understand the spatial distribution of criminal activities.

In summary, the image processing model for pistol detection offers a powerful tool for law enforcement agencies and security personnel to improve public safety. By harnessing the capabilities of advanced computer vision, this project underscores the potential of technology to complement human efforts in crime prevention and maintaining secure environments. As the model evolves and is integrated into security infrastructures, it has the potential to make significant contributions to the safety and well-being of communities.

**Links for the project:**

GitHub contains .ipynb file of our code :-

https://github.com/likhith530/ByteMasters

YouTube Link of Image Processing:-

https://www.youtube.com/watch?v=79gmSOcaVoE

YouTube Link of KMeans :-

https://www.youtube.com/watch?v=pjUSOcC5Rd0