Crime Vision: Advanced Crime Classification With Deep Learning

1.INTRODUCTION

1.1.OVERVIEW

Crime identification using deep learning is a technique that involves applying deep learning techniques, specifically deep learning, to analyze images and video footage of crime scenes or incidents and identify and classify different types of crimes. Deep learning involves training neural networks on large amounts of data to recognize patterns and make predictions or decisions. By using deep learning, it is possible to analyze images and video footage of crime scenes or incidents and classify different types of crimes based on the type of activity depicted in the images. This can be useful in a variety of criminal justice and law enforcement contexts, including crime scene investigation, forensic analysis, and surveillance.

Deep learning algorithms can be trained to recognize patterns and features in images and video that are relevant to identifying different types of crimes. They can also be used to analyze large amounts of data, such as surveillance footage, to identify trends and patterns in crime data. This can allow law enforcement agencies to develop strategies and interventions to prevent crime.

1.2.PURPOSE

This application of deep learning used in crime prediction is computer vision and video analysis. This technology has been used to analyze video footage from surveillance cameras to detect and classify criminal activities, such as vandalism, theft, and assault.

Here are some of the ways it can be utilized:

- Crime Detection
- Real-Time Crime Monitoring
- Evidence Collection
- Traffic Violation Detection
- Security Systems Enhancement
- Forensic Analysis
- Security Systems Enhancement
- Crime Pattern Recognition
- Social Media and Internet Monitoring

2.LITERATURE SURVEY

2.1.EXISTING PROBLEM

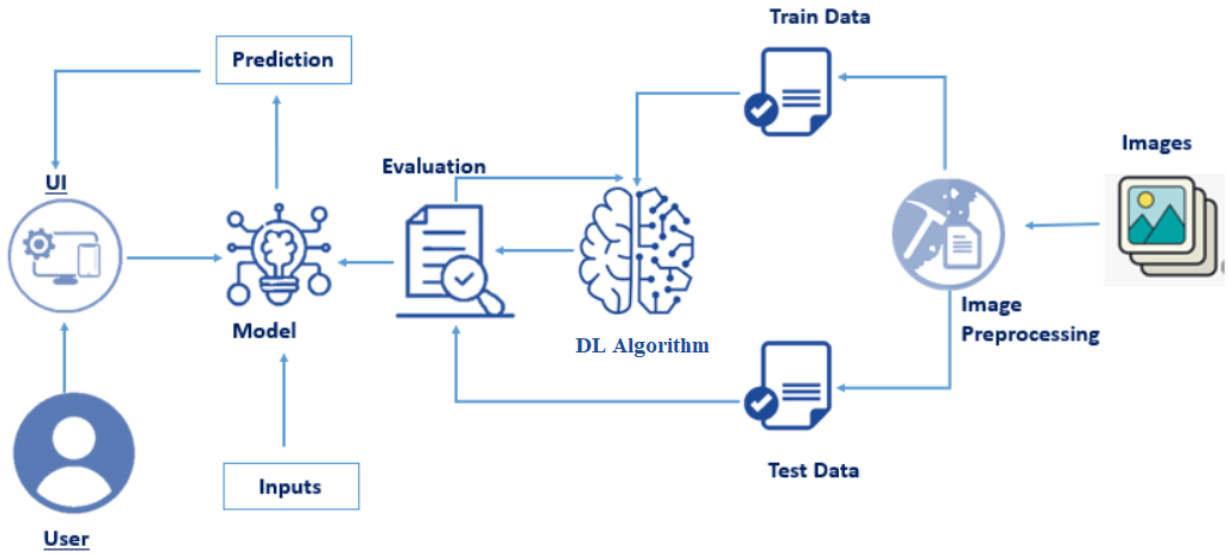
Here the existing problems are Implementing surveillance systems and collecting visual data for crime detection can raise significant privacy concerns. Balancing security needs with individual privacy rights is an ongoing challenge. The effectiveness of deep learning models heavily depends on the quality and quantity of the training data. Obtaining diverse, representative, and annotated data for all possible crime scenarios can be a significant challenge. Deep learning models can inherit biases present in training data. This can result in unfair or discriminatory outcomes, particularly in the identification of suspects. Ensuring fairness and minimizing bias is a critical problem Processing live video feeds in real-time requires powerful hardware and efficient algorithms. Latency issues can affect the ability to respond quickly to ongoing criminal activities. Implementing and maintaining advanced surveillance systems, including the necessary hardware and software, can be expensive. Reducing costs while maintaining effectiveness is an ongoing challenge. Recognizing unusual or novel criminal activities, rather than predefined categories, can be challenging. Anomaly detection in crowded scenes or complex environments is still an evolving area.

2.2.PROPOSED SOLUTION

Implement strict privacy policies and guidelines to ensure that surveillance and data collection are conducted with respect for individual privacy rights. Anonymize and encrypt data to protect the identities of innocent individuals. Collaborate with law enforcement agencies, security companies, and other stakeholders to share data and create comprehensive datasets for training and testing. Optimize hardware and software for real-time processing by utilizing GPUs, TPUs, and efficient deep learning models. Employ edge computing and distributed processing to reduce latency. Develop models that are robust to varying lighting conditions, weather, and camera quality. Utilize transfer learning and domain adaptation techniques to adapt to different environments. Explore advanced image enhancement and denoising techniques to improve the quality of low-light images. Use infrared and thermal imaging where applicable. Standardize communication protocols and data formats for interoperability between different surveillance systems. Encourage the use of open standards for better compatibility. Optimize algorithms and hardware for energy efficiency to reduce power consumption. Utilize energy-efficient processors and hardware accelerators.

3.THEORITICAL ANALYSIS

3.1.BLOCK DIAGRAM



3.2 HARDWARE / SOFTWARE DESIGNING

To complete this project, you must required the following software's, concepts and packages

1. Anaconda navigator and pharm
2. Python packages:

- Open anaconda prompt as administrator
- Type "pip install numpy" and click enter
- Type "pip install pandas" and click enter.
- Type "pip install Keras" and click enter.
- Type "pip install matplotlib" and click enter.
- Type "pip install TensorFlow" and click enter.
- Type "pip install Flask" and click enter.

4.EXPERIMENTAL INVESTIGATIONS

Experiments to evaluate the performance, accuracy, and reliability of deep learning models and computer vision systems for crime classification and detection. These experiments can provide insights into the practical feasibility of using advanced technologies in real-world crime prevention and law enforcement. Here are some areas where experimental investigations are commonly conducted:

- Experiment with various deep learning architectures, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and hybrid models, to assess their performance in crime classification tasks.
- Compare different pre-trained models and fine-tuning techniques for crime vision.
- Evaluate the quality and representativeness of crime-related datasets.

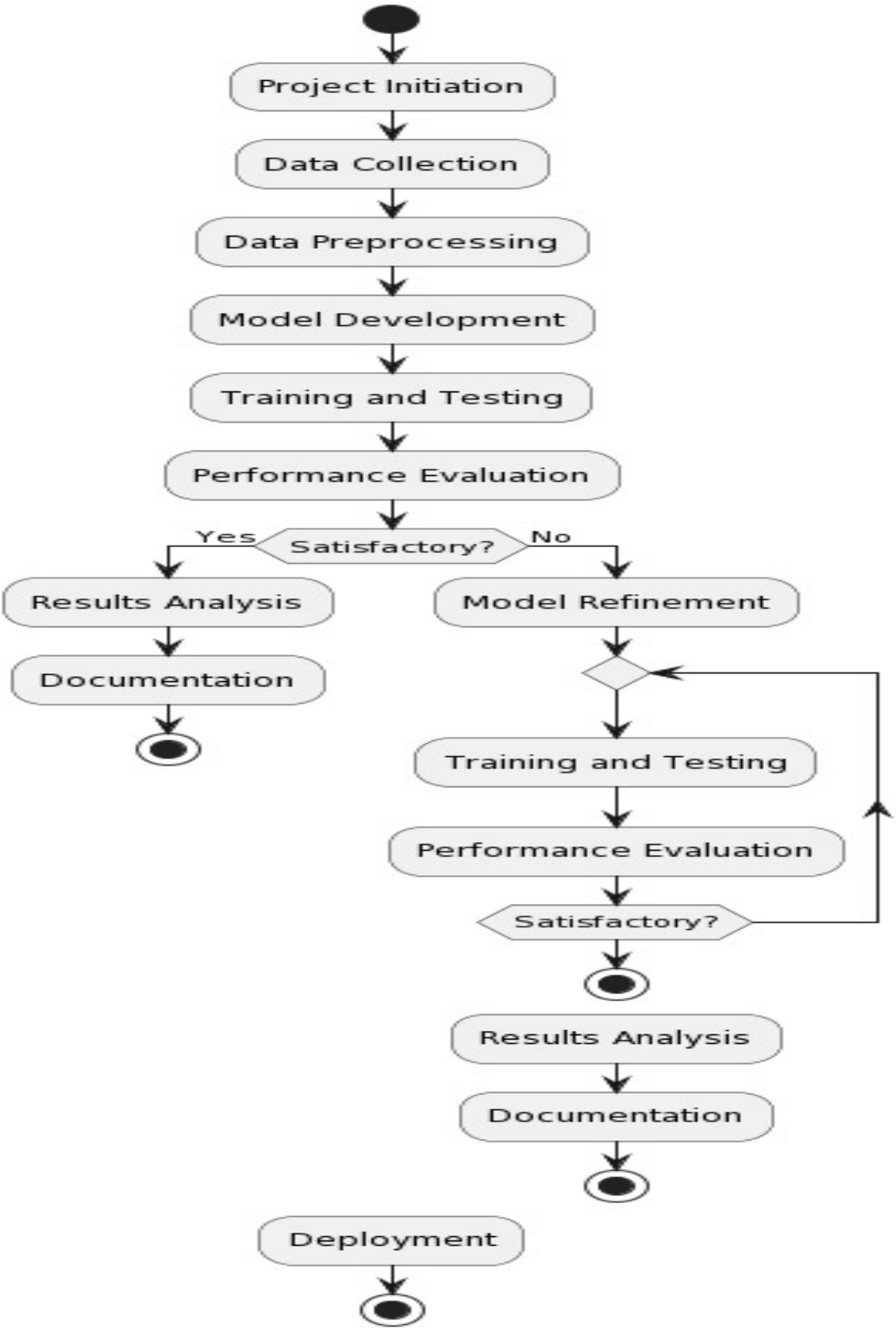
Project Flow:

- The user interacts with the UI to choose an image.
- model.The chosen image is processed by a Xception deep learning
- The Xception model is integrated with a Flask application.
- The Xception model analyzes the image and generates predictions.
- The predictions are displayed on the Flask UI for the user to see.
- This process enables users to input an image and receive accurate predictions quickly.

To accomplish this, we have to complete all the activities and tasks listed below

- Data Collection.
  - Create a Train and Test path.
- Image Pre-processing.
  - Import the required library
  - Configure ImageDataGenerator class
  - Apply functionality to Train set and Test set
- Model Building
  - Create Transfer Learning Function
  - Adding Dense Layer
  - Configure the Learning Process
  - Train the model
  - Save the Model
  - Test the model
- Application Building
  - Building HTML Pages
  - Building Flask Code
  - Run Application

5. FLOWCHART


Project Initiation
Data Collection
Data Preprocessing
Model Development
Training and Testing
Performance Evaluation
Satisfactory?
Yes
No
Results Analysis
Documentation
Model Refinement
Training and Testing
Performance Evaluation
Satisfactory?
Results Analysis
Documentation
Deployment

6. RESULT

Activity 1: Importing the libraries

Import the necessary libraries as shown in the image.

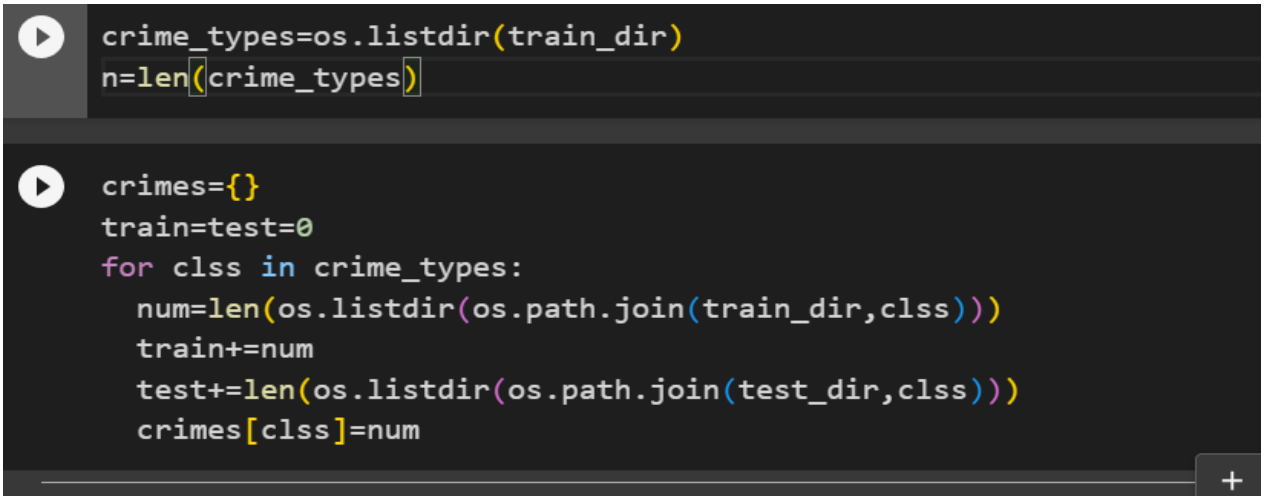
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import os
```

```
import tensorflow as tf
from tensorflow.keras.preprocessing import image_dataset_from_directory
```

```
from tensorflow.keras. applications import DenseNet121
from sklearn.preprocessing import LabelBinarizer
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout, MaxPooling2D, Conv2D, Flatten
from tensorflow.keras.models import Sequential
```

```
from IPython.display import clear_output
import warnings
warnings.filterwarnings('ignore')
```

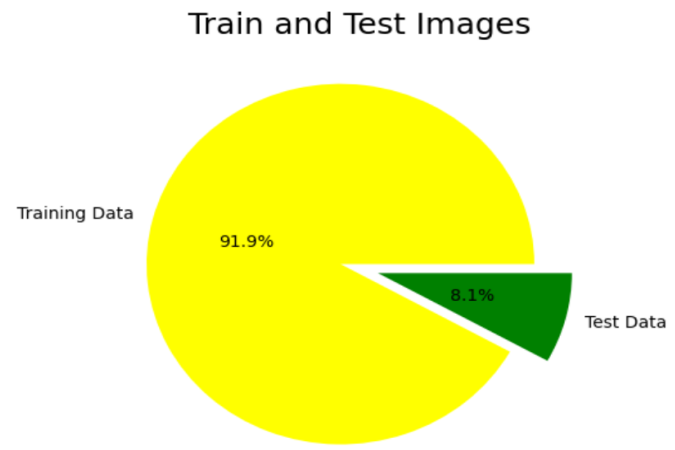
Activity 2: Read the Dataset



Activity 3: Train and Test Images

```
plt.figure(figsize=(15, 5))
plt.pie(x=np.array([train, test]), autopct="%.1f%%", explode=[0.1, 0.1], labels=["Training Data", "Test Data"], pctdistance=0.5,
plt.title("Train and Test Images", fontsize=18)
```

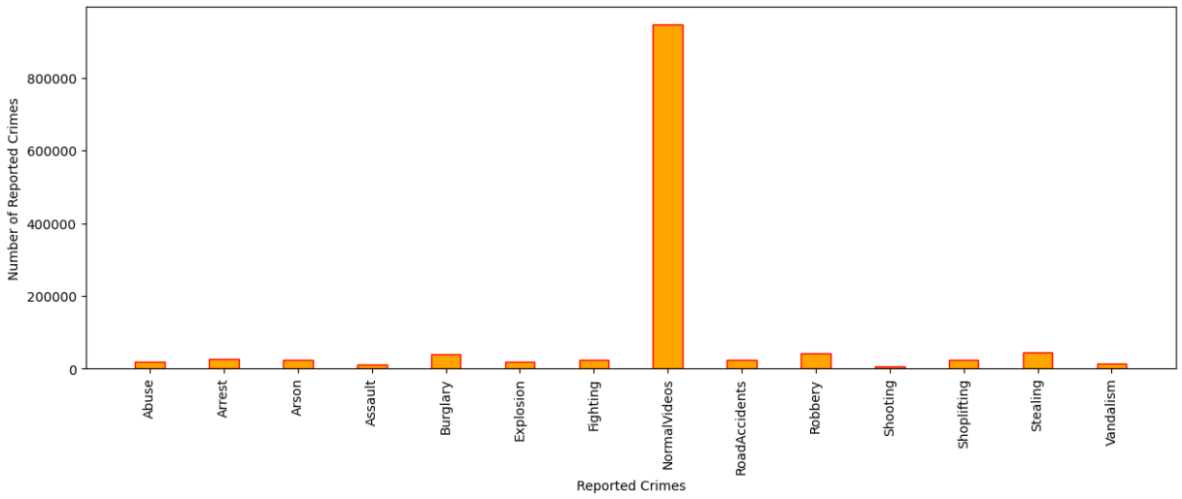
```
Text(0.5, 1.0, 'Train and Test Images')
```



Activity 4: crime analysis

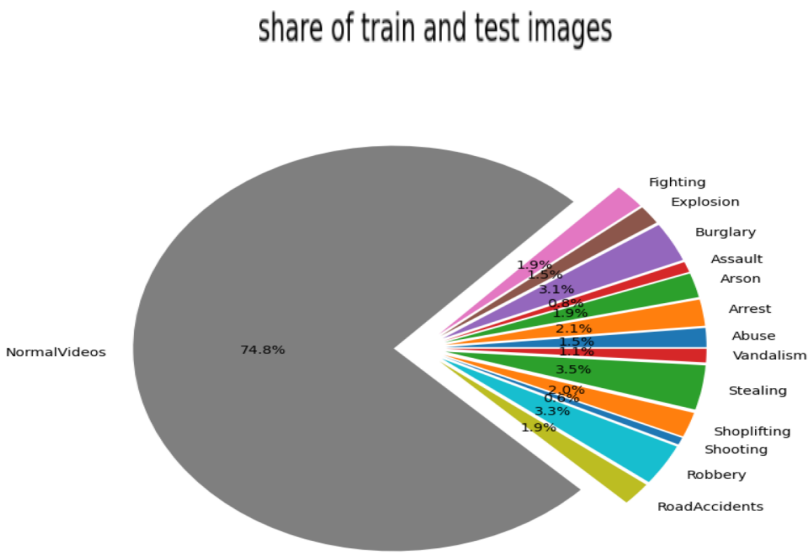
```
plt.figure(figsize=(15,5))
plt.bar(list(crimes.keys()), list(crimes.values()), width=0.4, align="center", edgecolor=['red'],color=['orange'])
plt.xticks(rotation=90)

plt.xlabel("Reported Crimes")
plt.ylabel("Number of Reported Crimes")
plt.show()
```



Activity 5: share of train and test images

```
plt.figure(figsize=(8,12))
plt.pie(x=np.array(list(crimes.values())), autopct="%.1f%%", explode=[0.1]*n, labels=list(crimes.keys()),pctdistance=0.5)
plt.title("share of train and test images", fontsize=14);
```



Activity 6: Apply Image_Dataset_from_directory Functionality To Train Set And Test Set

```
train_set=image_dataset_from_directory(train_dir,
                                       label_mode="categorical",
                                       batch_size=batch_size,
                                       image_size=(224,224),
                                       shuffle=True,
                                       seed=Seed,
                                       validation_split=0.2,
                                       subset="training",
                                       )
```

```
Found 1266345 files belonging to 14 classes.
Using 1013076 files for training.
```

+ Code + Text

```
[14] val_set=image_dataset_from_directory(train_dir,
                                       label_mode="categorical",
                                       batch_size=batch_size,
                                       image_size=(224,224),
                                       shuffle=True,
                                       seed=Seed,
                                       validation_split=0.2,
                                       subset="validation",
                                       )
```

```
Found 1266345 files belonging to 14 classes.
Using 253269 files for validation.
```

```
test_set=image_dataset_from_directory(test_dir,
                                      label_mode="categorical",
                                      class_names=None,
                                      batch_size=batch_size,
                                      image_size=(224,224),
                                      shuffle=False,
                                      seed=Seed,
                                      )
```

```
Found 111308 files belonging to 14 classes.
```

Activity 7: Create Transfer Learning Function

```
def transfer_learning():
    base_model=DenseNet121(include_top=False,input_shape=INPUT_SHAPE,weights="imagenet")

    thr=149
    for layers in base_model.layers[:thr]:
        layers.trainable=False

    for layers in base_model.layers[thr:]:
        layers.trainable=True

    return base_model
```

```
model = create_model()

model.compile(optimizer="adam", loss='categorical_crossentropy', metrics = ['accuracy'])
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 densenet121 (Functional)    (None, 2, 2, 1024)        7037504

 global_average_pooling2d (  (None, 1024)              0
 GlobalAveragePooling2D)

 dense (Dense)               (None, 256)               262400

 dropout (Dropout)           (None, 256)               0

 dense_1 (Dense)             (None, 512)               131584

 dropout_1 (Dropout)         (None, 512)               0

 dense_2 (Dense)             (None, 1024)              525312

 classification (Dense)      (None, 14)                14350

=================================================================
Total params: 7971150 (30.41 MB)
Trainable params: 6386894 (24.36 MB)
Non-trainable params: 1584256 (6.04 MB)
```

Activity 8: Train The Model

```
history=model.fit(x=train_set,validation_data=val_set,epochs=epochs)
```

```
Epoch 1/5
7915/7915 [==============================] - 3595s 451ms/step - loss: 0.2922 - accuracy: 0.9161 - val_loss: 0.0955 - val_accuracy: 0.9732
Epoch 2/5
7915/7915 [==============================] - 3587s 453ms/step - loss: 0.1484 - accuracy: 0.9570 - val_loss: 0.0537 - val_accuracy: 0.9860
Epoch 3/5
7915/7915 [==============================] - 3590s 453ms/step - loss: 0.1245 - accuracy: 0.9647 - val_loss: 0.0418 - val_accuracy: 0.9888
Epoch 4/5
7915/7915 [==============================] - 3594s 454ms/step - loss: 0.1128 - accuracy: 0.9683 - val_loss: 0.0358 - val_accuracy: 0.9905
Epoch 5/5
7915/7915 [==============================] - 3603s 455ms/step - loss: 0.1063 - accuracy: 0.9708 - val_loss: 0.0349 - val_accuracy: 0.9908
```

Activity 9: Save The Model

```
model.save('crime.h5')
```

```
from tensorflow.keras.models import load_model
model.load_weights('crime.h5')
```

```
y_true = np.array([])

for x,y in test_set:
    y_true = np.concatenate([y_true, np.argmax(y.numpy(), axis=-1)])
```

Activity 10: testing

```
#testing 1
img = image.load_img('/content/dataset/Test/RoadAccidents/RoadAccidents001_x264_10.png',target_size=(64,64)) #Reading image
x = image.img_to_array(img) # converting image into array
x = np.expand_dims(x,axis=0) # expanding Dimensions
pred = np.argmax(model.predict(x)) #Predicting the higher probability index
op = ['Fighting','Arrest','Vandalism','Assault','Stealing','Arson','NormalVideos','Burglary','Explosion','Robbery','Abuse','Shooting','Shoplifting','
op[pred] #List indexing with output
```

```
1/1 [==============================] - 2s 2s/step
'RoadAccident'
```

```
#testing 2
img = image.load_img('/content/dataset/Test/Explosion/Explosion002_x264_10.png',target_size=(64,64)) #Reading image
x = image.img_to_array(img) # converting image into array
x = np.expand_dims(x,axis=0) # expanding Dimensions
pred = np.argmax(model.predict(x)) #Predicting the higher probability index
op = ['Fighting','Arrest','Vandalism','Assault','Stealing','Arson','NormalVideos','Burglary','Explosion','Robbery','Abuse','Shooting','Shoplifting','
op[pred] #List indexing with output
```

```
1/1 [==============================] - 3s 3s/step
'Explosion'
```

```
#testing 3
img = image.load_img('/content/dataset/Test/Abuse/Abuse028_x264_10.png',target_size=(64,64)) #Reading image
x = image.img_to_array(img) # converting image into array
x = np.expand_dims(x,axis=0) # expanding Dimensions
pred = np.argmax(model.predict(x)) #Predicting the higher probability index
op = ['Fighting','Arrest','Vandalism','Assault','Stealing','Arson','NormalVideos','Burglary','Explosion','Robbery','Abuse','Shooting','Shoplifting','
op[pred] #List indexing with output
```

```
1/1 [==============================] - 0s 73ms/step
'Abuse'
```

Activity 11: Application Building

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

This section has the following tasks

12. Building HTML Pages
13. Building server side script

Activity12: Building Html Pages:

For this project create two HTML files namely

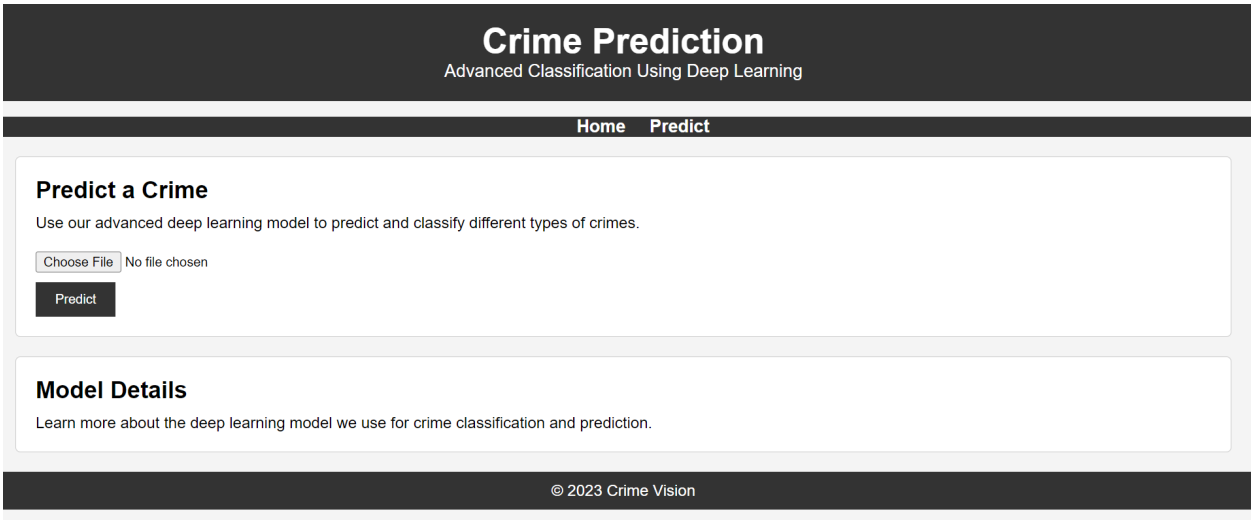
1.Home.html

2. predict.html and save them in templates folder.

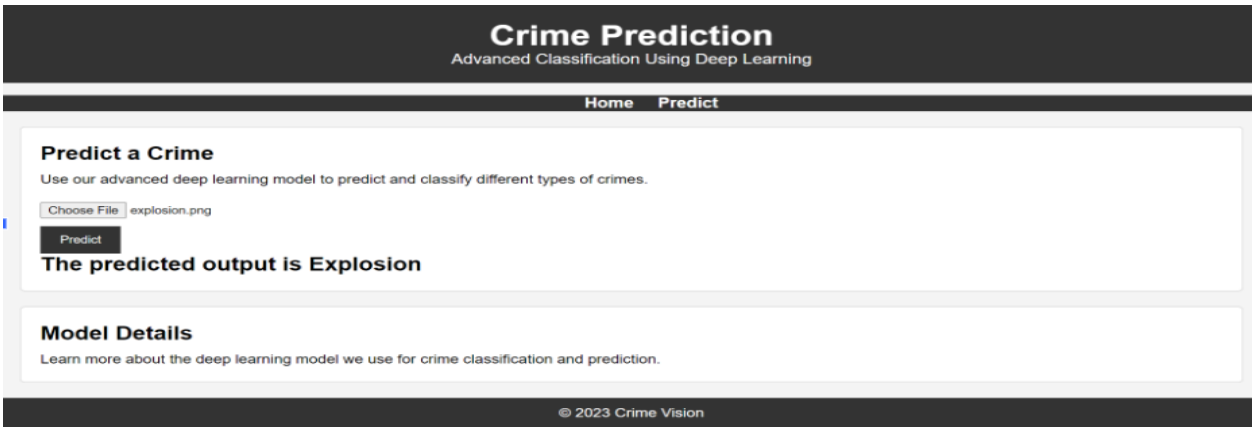
Let’s see how our index.html page looks like:

INDEX.HTML:

PREDICT.HTML:



Output:



7. ADVANTAGES & DISADVANTAGE

Advantages:

1. Improved Accuracy: Deep learning algorithms can analyze large datasets and make predictions with high accuracy, potentially reducing the misclassification of crimes.
2. Automation: The system can automate the crime classification process, reducing the workload on human law enforcement personnel and potentially speeding up investigations.
3. Speed: Deep learning models can process and classify data quickly, which can help law enforcement respond to crimes faster.
4. Scalability: The system can handle a large volume of data and can be scaled up to process information from various sources, such as security cameras, social media, and more.
5. Pattern Recognition: Deep learning can identify subtle patterns and relationships in crime data that may not be apparent to human analysts, potentially leading to better insights.
6. Consistency: Unlike human analysts, the system can provide consistent results and is not affected by fatigue or bias.

Disadvantages:

1. Data Bias: If the training data used to develop the deep learning model is biased, it can lead to unfair and discriminatory results, potentially reinforcing existing biases in the criminal justice system.
2. Lack of Context: Deep learning models may not fully understand the context of a crime, which can lead to misclassification or errors in understanding the severity of a situation.
3. Data Privacy Concerns: Analyzing data from various sources, including surveillance cameras and social media, may raise privacy concerns and potential violations of civil liberties.

4. Cost and Resource Intensity: Developing and maintaining deep learning systems can be expensive and resource-intensive, requiring significant computational power and expertise.
5. Ethical Concerns: The use of advanced technology in crime classification raises ethical questions, particularly around issues like surveillance, consent, and accountability.
6. Limited Training Data: Obtaining a sufficient amount of diverse and representative training data for crime classification can be challenging, potentially limiting the model's effectiveness.

8.APPLICATIONS

1. Deep Learning Frameworks: Deep learning models for crime classification can be built using popular frameworks such as TensorFlow, PyTorch, or Keras. These frameworks provide tools and libraries for creating, training, and deploying deep neural networks.
2. Data Collection and Integration: An application for crime classification would require the integration of various data sources, including crime reports, surveillance footage, social media data, and more. APIs, data connectors, and data preprocessing techniques would be used for data collection and integration.
3. Neural Networks: Deep learning models, such as convolutional neural networks (CNNs) for image analysis or recurrent neural networks (RNNs) for text analysis, would be used for crime classification tasks. These networks are designed to extract features and patterns from different types of data.
4. Training Data: High-quality and diverse training data are crucial for training deep learning models. This data would consist of labeled examples of different types of crimes.
5. Feature Extraction: Preprocessing and feature extraction techniques may be applied to the raw data to prepare it for input to the deep learning models. For example, image data might be resized and normalized, and text data might be tokenized and vectorized.
6. Model Training: Deep learning models would be trained on the prepared data to learn patterns and features related to different crime types. Training involves multiple iterations and adjustments to optimize model performance.
7. Model Evaluation: The trained models would need to be evaluated using metrics like accuracy, precision, recall, and F1-score to assess their performance. This step helps fine-tune the models and ensures they meet the desired accuracy and reliability standards.
8. Deployment: Once trained and evaluated, the models are deployed as part of the application, where they can classify crimes in real-time or analyze historical data.
9. User Interface: The application would typically have a user interface for law enforcement personnel to interact with the system, view results, and access insights generated by the deep learning models.

10. Security and Privacy Measures: Robust security and privacy measures are essential to protect the sensitive data involved in crime classification. This includes encryption, access control, and compliance with data protection regulations.
11. Regular Updates and Maintenance: Continuous updates, monitoring, and maintenance are necessary to keep the application running effectively, adapt to changing crime patterns, and address issues as they arise

9.CONCLUSION

In conclusion, the concept of "Crime Vision" highlights the potential for advanced technologies to enhance crime classification and law enforcement efforts. However, the development and implementation of such a system would need to address ethical, privacy, and fairness concerns while ensuring transparency, accountability, and compliance with legal regulations. The benefits of accuracy, efficiency, and speed must be balanced against the challenges of bias, privacy, and the potential for misuse. The responsible use of technology in crime classification is crucial to maintain trust in the criminal justice system and protect the rights and privacy of individuals.

10.FUTURE SCOPE

In summary, the future scope of a project like "Crime Vision" is bright, with potential for improving crime classification, law enforcement efficiency, and public safety. However, it must be accompanied by responsible development and a strong commitment to addressing ethical, privacy, and security concerns in the application of advanced technology to the criminal justice system.

1. Improved Model Performance: As deep learning models continue to evolve, the future scope involves achieving even higher accuracy in crime classification. This would involve developing more sophisticated neural network architectures and refining training techniques.
2. Multimodal Data Integration: Future iterations of the project may expand to incorporate a wider variety of data sources, including audio, sensor data, and biometrics, allowing for more comprehensive crime analysis and classification.
3. Real-time Crime Detection: Enhancing the project to provide real-time crime detection and alerts would be a valuable development. This could assist law enforcement in responding more quickly to criminal activities.

11.BIBILOGRAPHY

- www.google.com
- www.wikipedia.org
- https://ieeexplore.ieee.org

12.APPENDIX

SOURCE CODE OF FLASK

https://github.com/smartinternz02/SI-GuidedProject-589413-1697085022