

Hiding Sensitive Data from Images and Generic Documents

supervised by

Prasun Agarwal
Micron Technologies

Boda Surya Venkata
Jyothi Sowmya
MSc. Data Science, CMI

cmi



Presentation Outline

💡 Introduction

🔍 Problem Statement

💭 Motivation for this Project

🔗 Project Pipeline

📊 Data Description

📄 ID Detection

✂ Cropping

📏 Alignment

📄 Text Extraction

🎭 PII Field Detection &
Masking

🧠 Streamlit App

📺 Results & Evaluation

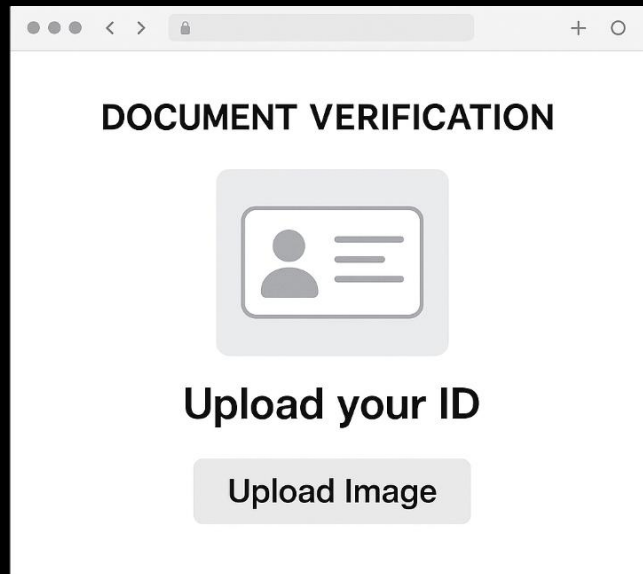
⚠ Challenges Faced

🚀 Future Scope

⌚ Conclusion

Introduction

- Many people share ID cards or bills online for verification or services
- These documents often contain PII details (Personally Identifiable Information) like names, ID numbers, phone numbers or addresses.
- This project builds an app that automatically hides sensitive data in images.
- **Goal:** Protect privacy while keeping documents usable



Problem Statement

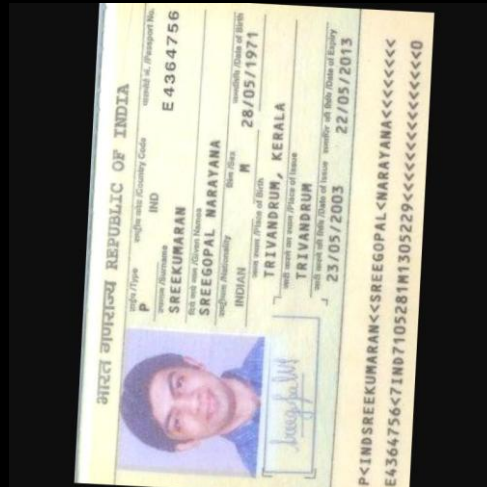
- **Problem** : Sharing documents like Aadhaar, PAN cards, or passports digitally can expose sensitive personal information. Manually hiding these details is time-consuming, inconsistent, and prone to human error
- **Risk** : Exposed information can be leaked or misused, leading to risks such as identity theft, fraud or data breaches
- **Challenge** : Documents come in various formats and may be captured under different lighting, angles, or backgrounds, making the detection harder.



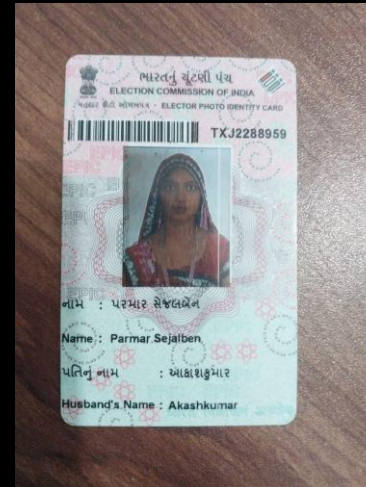
Driving Licence
with background



Tilted Pan card



Tilted Passport

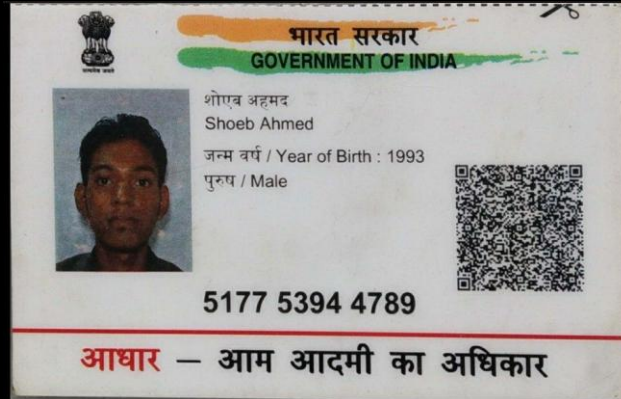


Voter ID with background



Tilted Aadhaar Card

Original Image



Masked Image



- **Objective** : Build an app that can automatically detect, extract, and mask only sensitive data from images.
- **Outcome** : The masked document retains all non-sensitive parts, so it can still be shared and used for verification – without risking privacy

Motivation for this Project

Everyday documents like Aadhaar cards, passports or water bills carry names, ID numbers and addresses. If these leak, anyone can misuse them for identity theft or fraud.

Real-World Examples :

- **Tamil Nadu's PDS Data Allegedly Breached**

Source : <https://www.gadgets360.com/internet/news/tamil-nadu-pds-system-data-breach-50-lakh-aadhaar-card-numbers-leak-technisanct-2475988>

- **Aadhaar Data Out of Closet, Educational Institutes Guilty of Leak**

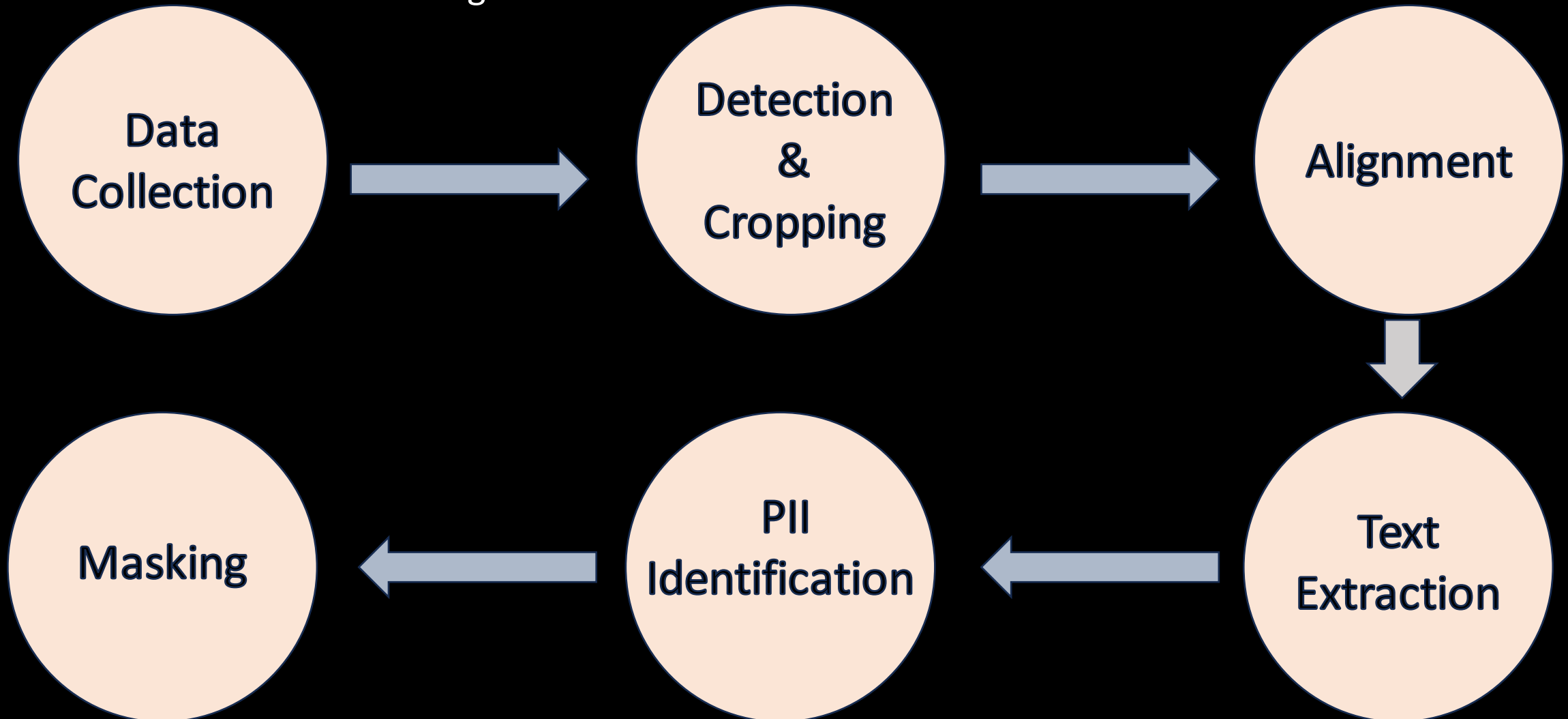
Source : <https://timesofindia.indiatimes.com/city/hyderabad/aadhaar-data-out-of-closet-educational-institutes-guilty-of-leak/articleshow/61732743.cms>

- **Massive Aadhaar Data Spill in 2018**

Source : <https://www.bbc.com/news/world-asia-india-42575443>

Project Pipeline

This is a six-stage process that combines computer vision and an LLM to automatically detect and hide sensitive data in document images.



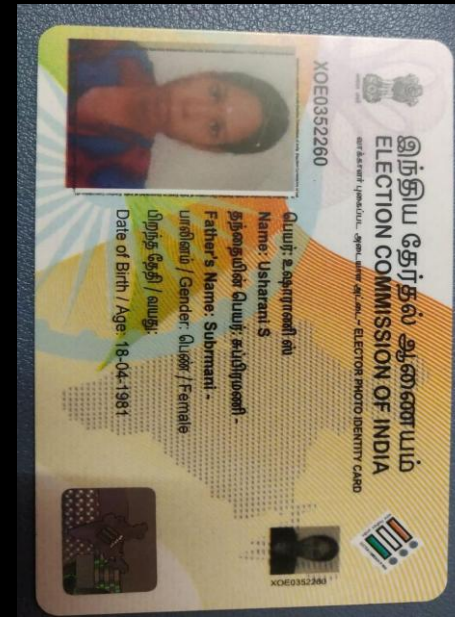
Flow Chart



Data Collection



Card Detection



Cropped



Alignment



1: 65i00 1.000
2: ELECTION COMMISSION OFINDIA 1.000
3: LL 1.000
4: ELECTOR PHOTO IDENTITY CARD 1.000
5: XOE0352260 1.000
6: XOE03522G0 1.000
7: Name 1.000
8: Usharani S 1.000
9: Father's Name 1.000
10: :Subramani 1.000
11: Llr6r/GenderOu6rFemale 1.000
12: Lmgg/g 1.000
13: Date of Birth 1.000
14: /Age:18-04-1981 1.000

Text Extraction



Masked

Dataset Description

Source: Publicly available datasets from Google and websites.

Document Types:

- Aadhaar Card
- Passport
- PAN Card
- Driving License
- Voter Card
- Delhi Water Bills

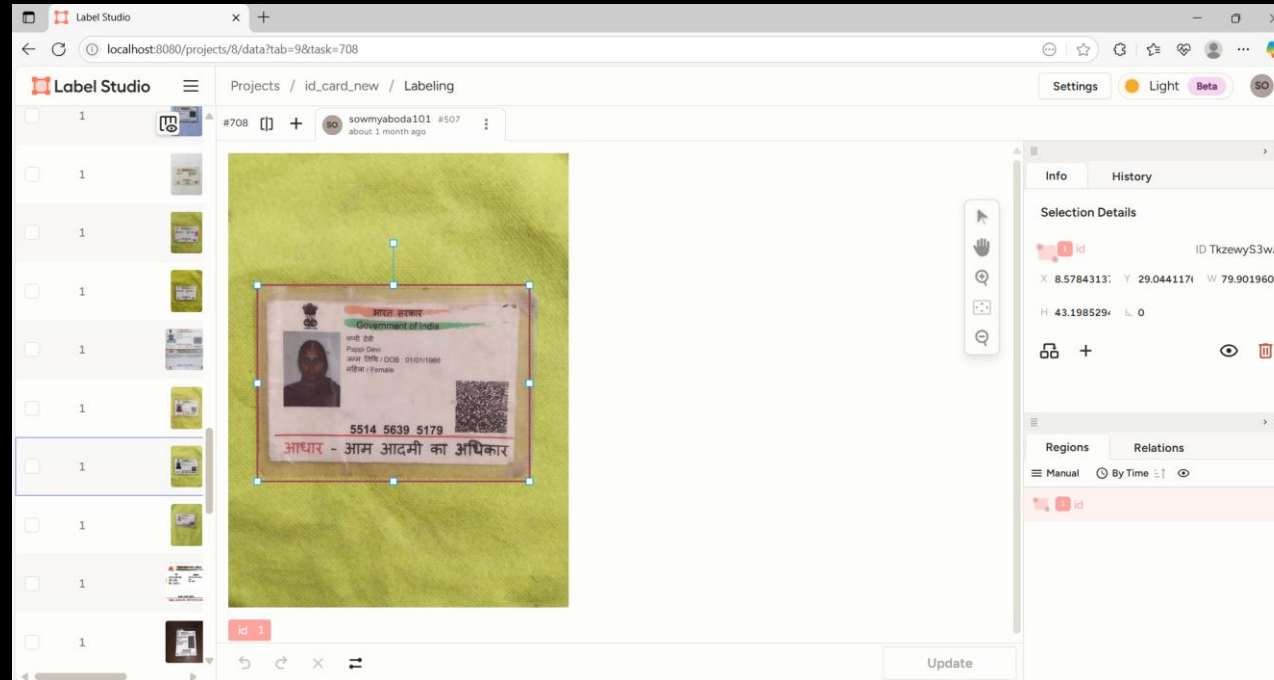
Characteristics:

- Varied backgrounds.
- Different lighting.
- Mixed orientations.

Size: Large set of diverse images.

Type of card	Trianing Data	Test Data
Aadhaar Card	153	12
Driving Licence	131	12
Pan Card	138	12
Passport	102	12
Voter Card	164	12
Water Bills	92	12
Total Data	780	72

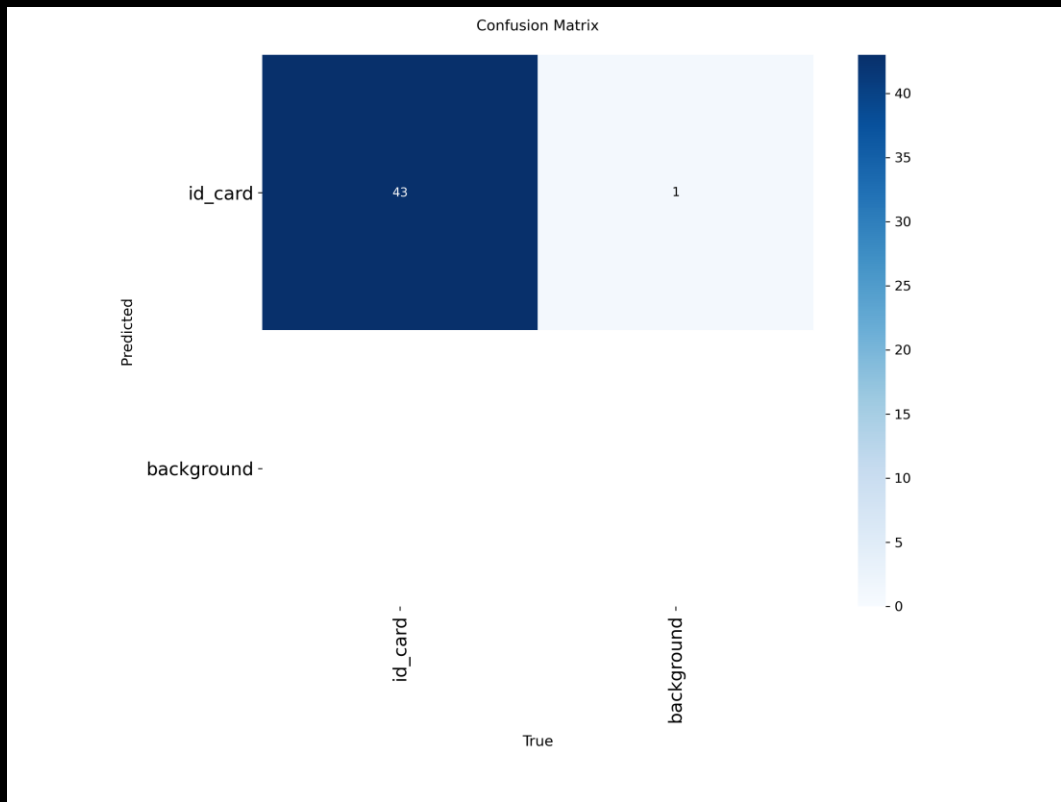
ID card Detection and Cropping



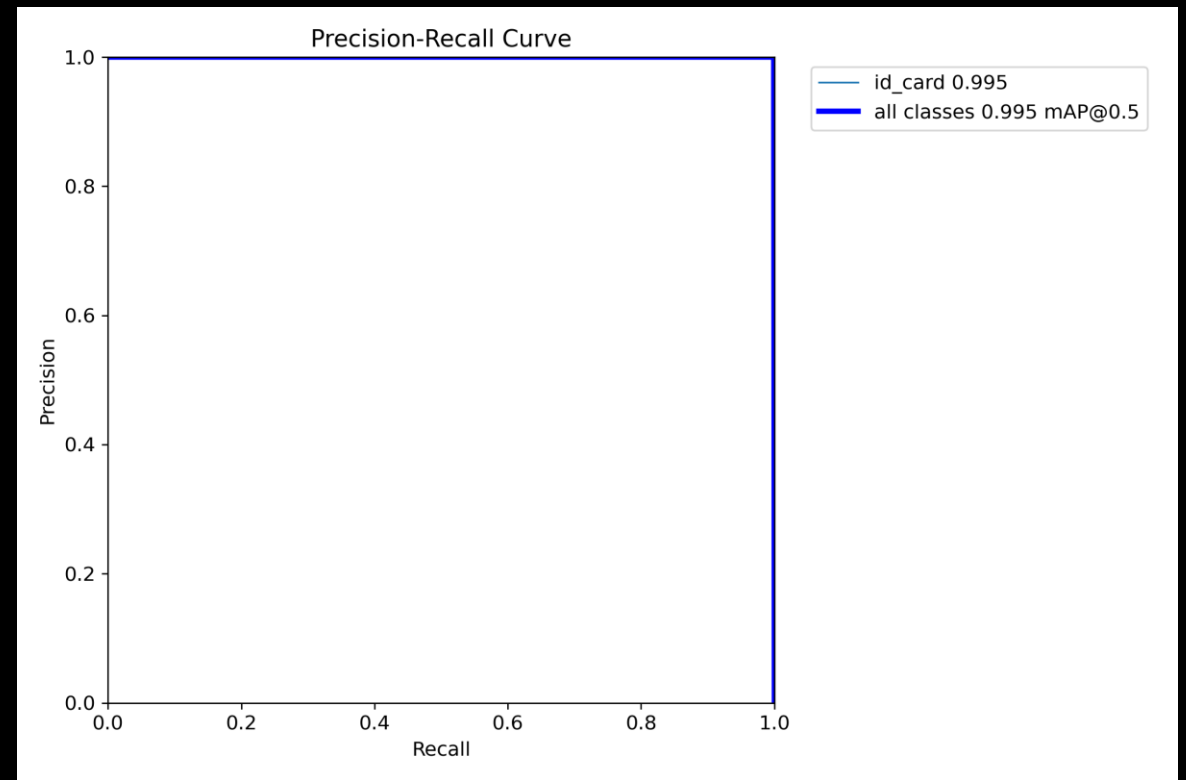
Tool : YOLOv8 (Object Detection Model).

Process :

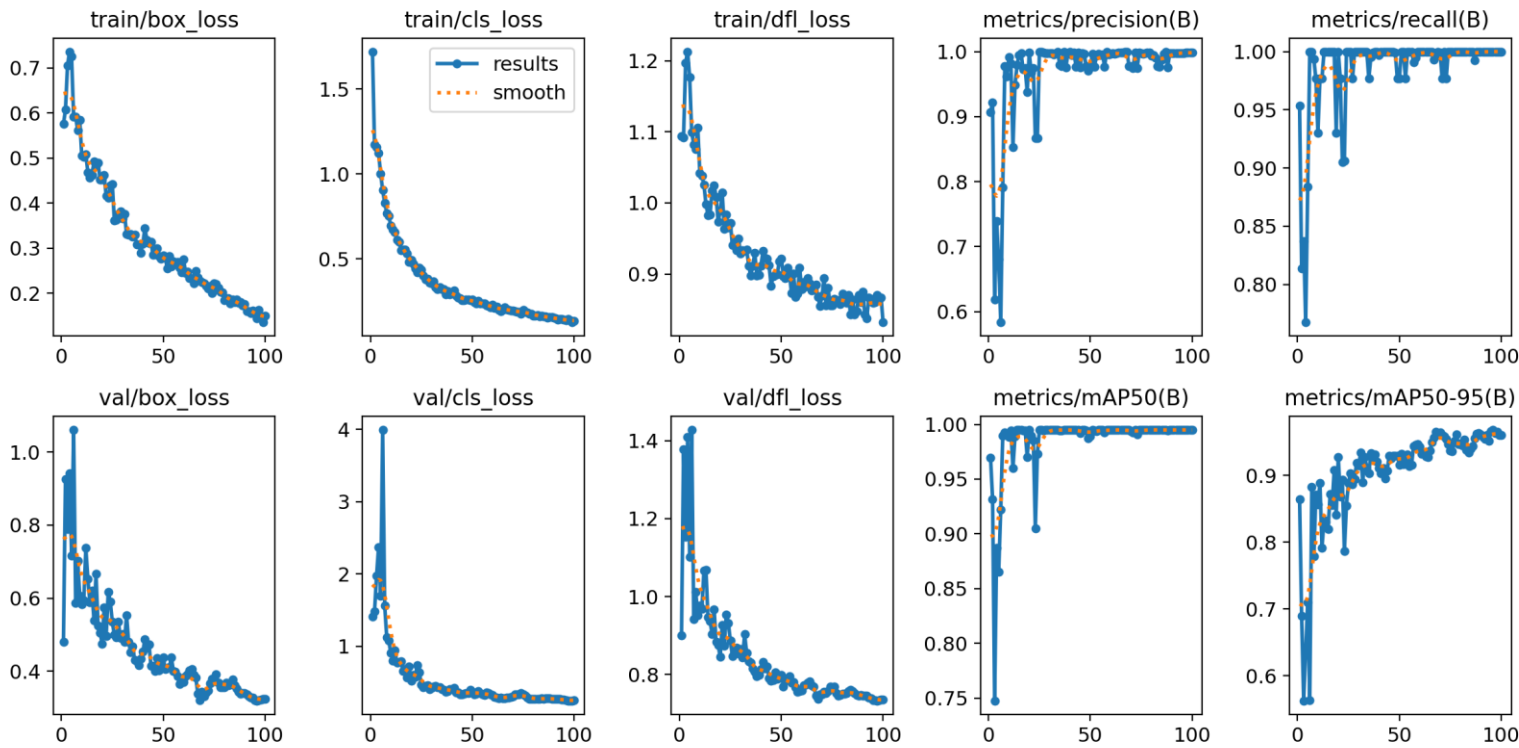
- Annotated images with bounding boxes around ID cards using LabelStudio (Open Source)
- Trained YOLOv8 on the labeled dataset to learn card detection
- Ran the trained model to predict bounding boxes on all images
- Cropped each detected card region for downstream processing



- The model correctly predicted id_card when it was actually id_card (43 TP)
- The model predicted id_card but it was actually background (for only 1)



- This graph shows how well the model finds and correctly identifies ID cards
- It has high precision and recall, so it finds most ID cards without any mistakes
- The score mAP@0.5 is 0.995 which means the model is 99.5% accurate in detecting ID cards.



- **train/box_loss** : Shows the model is learning to draw accurate boxes
- **train/cls_loss** : Classification loss is decreasing. So model recognizes ID cards
- **train/dfl_loss** : Low dfl_loss. Model refines box predictions more accurately
- **metrics/precision(B)** : High precision shows most predicted ID cards are correct
- **metrics/recall(B)** : High recall shows model finds most of the ID cards

- **val/box_loss** : The model works well on unseen data as it has lower box loss
- **val/cls_loss** : The model identifies ID cards correctly on test data (low validation classification loss)
- **val/dfl_loss** : Low dfl_loss. Model places boxes accurately even on validation images.
- **metrics/mAP50(B)** : Object detection is almost perfect at standard IoU threshold
- **metrics/mAP50-95(B)** : Model does well even with stricter accuracy conditions too.



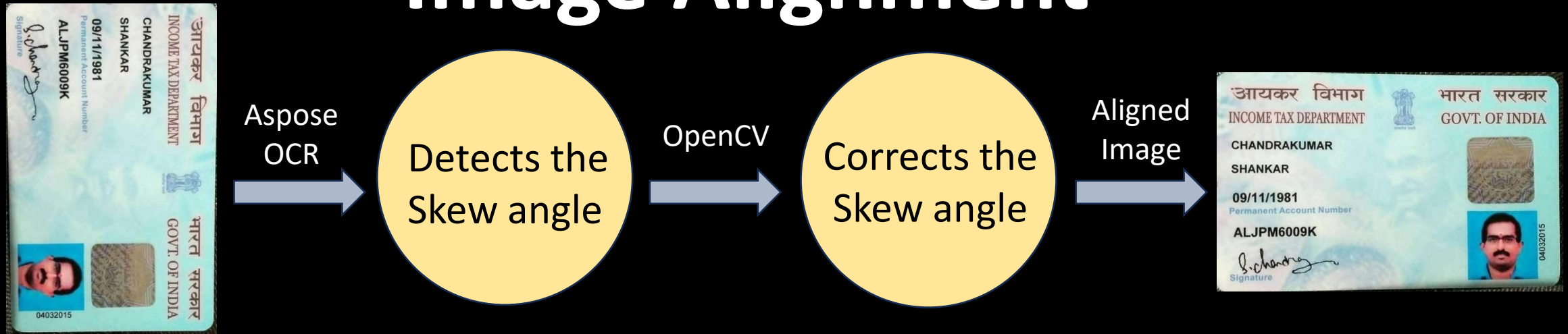
Cropping



Why YOLOv8?: Fast, accurate, handles diverse backgrounds/orientations.

Outcome : Isolated ID card regions for alignment and text extraction.

Image Alignment



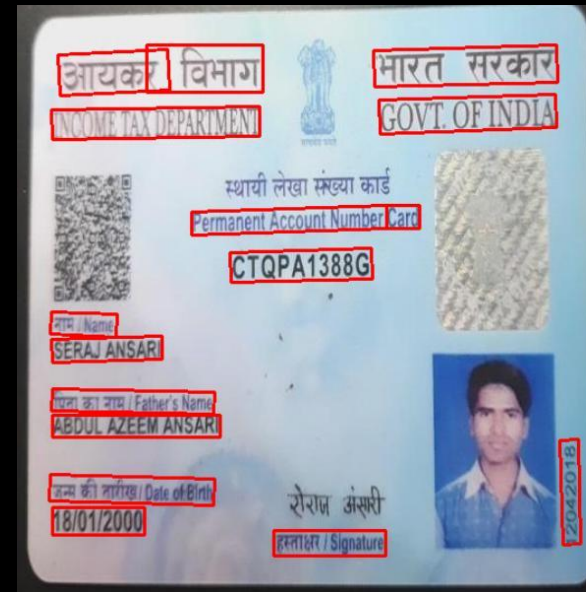
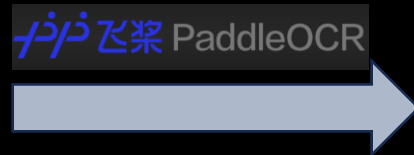
Tools : AsposeOCR (Skew Detection) & OpenCV (Rotation & Warping)

Process :

- Loaded the cropped images
- Calculated skew angle using AsposeOCR
- Rotate images using OpenCV's rotation matrix
 - Compute new dimensions to avoid cropping.
 - Use cv2.warpAffine for rotation.

Outcome : Aligned images for better text extraction.

Text Extraction



1: BITeTaT 1.000
2: TT 1.000
3: HR RR 1.000
4: INCOME TAX DEPARTMENT 1.000
5: GOVT.OF INDIA 1.000
6: Permanent Account Number 1.000
7: Card 1.000
8: CTQPA1388G 1.000
9: T/Name 1.000
10: SERAJ ANSARI 1.000
11: Father's Name 1.000
12: ABDULAZEEM ANSARI 1.000
13: 12042018 1.000
14: aleot Bi 1.000
15: 18/01/2000 1.000
16: T/Signature 1.000

Tools : PaddleOCR (Text Detection)

Process :

- Input the aligned images
- Extract the text and their bounding boxes

Outcome : Text list with coordinates for PII identification.

Tools Tested : Pytesseract, PaddleOCR, Azure OCR

Focus : Performance on poor-quality image

- Pytesseract :
 - Struggles with poor-quality images
 - Extracted text is often incomplete or inaccurate.
- PaddleOCR
 - Performs better than Pytesseract on poor-quality images.
 - Can detect more text regions and characters accurately.
 - Extracted text is **good enough** for **further enhancement** by LLMs (e.g., Gemini) for tasks like **PII extraction**.
- Microsoft Azure OCR
 - Offers the best accuracy for poor-quality images
 - Handles poor lighting, distortions, and noise effectively.
 - Limitation : Paid API, usage is restricted without a subscription



```

4
Goda: Carga querer
UN@ene FOR Rue AL a
114 CTION COMMISSION OF INDIA
IDENTITY CARD
A+ 504385632
we g
tae Gas amgGa
lect' & Matte Karta
aarp uu Upo@iann
+ athwe » Narte Padmenumsr
Wrz fee amen Maw
| 3am Cog $ Oene of Sete 21/07/1008
  
```

Pytesseract Result



```

1: G 1.000
2: 1.000
3: LECTION COMMISSION OFINDIA 1.000
4: IDENTITY-CARD 1.000
5: XKR1385632 1.000
6: innt Cuannod 1.000
7: Eintot Ne 1.000
8: Kartha 1.000
9: FatrNmPadaxumr 1.000
10: Mate 1.000
11: CD 21199 1.000
  
```

Paddle Result

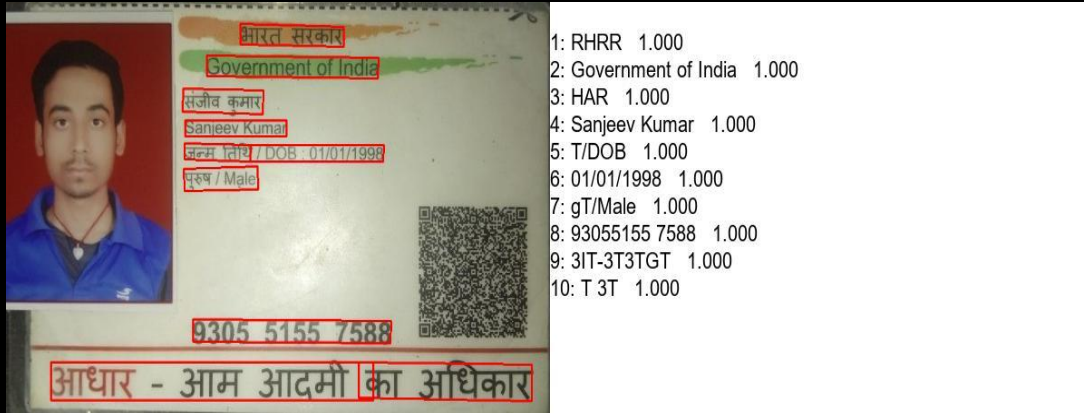


```

"text": "ELECTION COMMISSION OF INDIA"
"text": "IDENTITY CARD"
"text": "XKR1385632"
"text": "பிரச்சனைனா பெயர் கார்த்திக்"
"text": "Elpotit's NHitte"
"text": "Karthla"
"text": "தத்தை பெயா பத்மகுமார்"
"text": "Padmaxumur"
"text": "Male"
"text": "ose C+6 / Dein of Birth"
"text": "21/02/1993"
  
```

Microsoft Azure
Vision Result

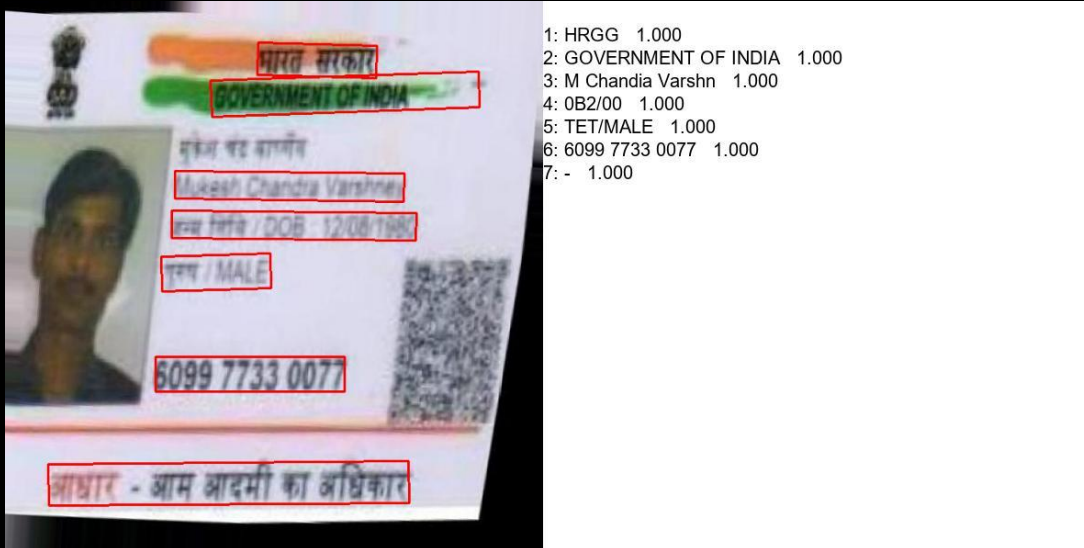
PII Field Extraction



- The classification of document type is performed within the same LLM prompt that detects PII — based on the combined context of OCR text and image.

```
{
  "Document Type": "Aadhaar Card",
  "Name": {
    "value": "Sanjeev Kumar",
    "coordinates": [
      [
        { "x": 272, "y": 149 },
        { "x": 426, "y": 149 },
        { "x": 426, "y": 169 },
        { "x": 272, "y": 169 }
      ]
    ]
  },
  "Date of Birth": {
    "value": "01/01/1998",
    "coordinates": [
      [
        { "x": 372, "y": 180 },
        { "x": 574, "y": 180 },
        { "x": 573, "y": 201 },
        { "x": 371, "y": 201 }
      ]
    ]
  },
  "Aadhaar ID": {
    "value": "930551557588",
    "coordinates": [
      [
        { "x": 284, "y": 400 },
        { "x": 585, "y": 402 },
        { "x": 584, "y": 430 },
        { "x": 284, "y": 428 }
      ]
    ]
  }
}
```

- The LLM improves low-quality OCR text by using the original image for better PII extraction, while relying on OCR annotations for accurate coordinates.



```
{
  "Document Type": "Aadhaar Card",
  "Name": {
    "value": "Mukesh Chandra Varshney",
    "coordinates": [
      [
        { "x": 210, "y": 210 },
        { "x": 494, "y": 213 },
        { "x": 493, "y": 248 },
        { "x": 210, "y": 244 }
      ]
    ]
  },
  "Date of Birth": {
    "value": "12/08/1980",
    "coordinates": [
      [
        { "x": 206, "y": 259 },
        { "x": 509, "y": 262 },
        { "x": 508, "y": 296 },
        { "x": 206, "y": 293 }
      ]
    ]
  },
  "Aadhaar ID": {
    "value": "609977330077",
    "coordinates": [
      [
        { "x": 186, "y": 440 },
        { "x": 420, "y": 440 },
        { "x": 420, "y": 483 },
        { "x": 186, "y": 483 }
      ]
    ]
  }
}
```

Tools : Gemini 1.5 Flash (LLM for Text Analysis)

Process :

- Input the PaddleOCR text to Gemini 1.5 Flash
- The LLM identifies PII data such as names, ID numbers, addresses, etc
- Match each identified PII item back to its corresponding coordinates from PaddleOCR

Why Gemini 1.5 Flash :

- Fast and accurate entity recognition
- Handles multiple document types with varying formats
- Free API access for development and testing

Outcome : A list of sensitive text fields along with their coordinates, ready for masking

PII Masking



Aligned Image



Masked Image

Tools : OpenCV

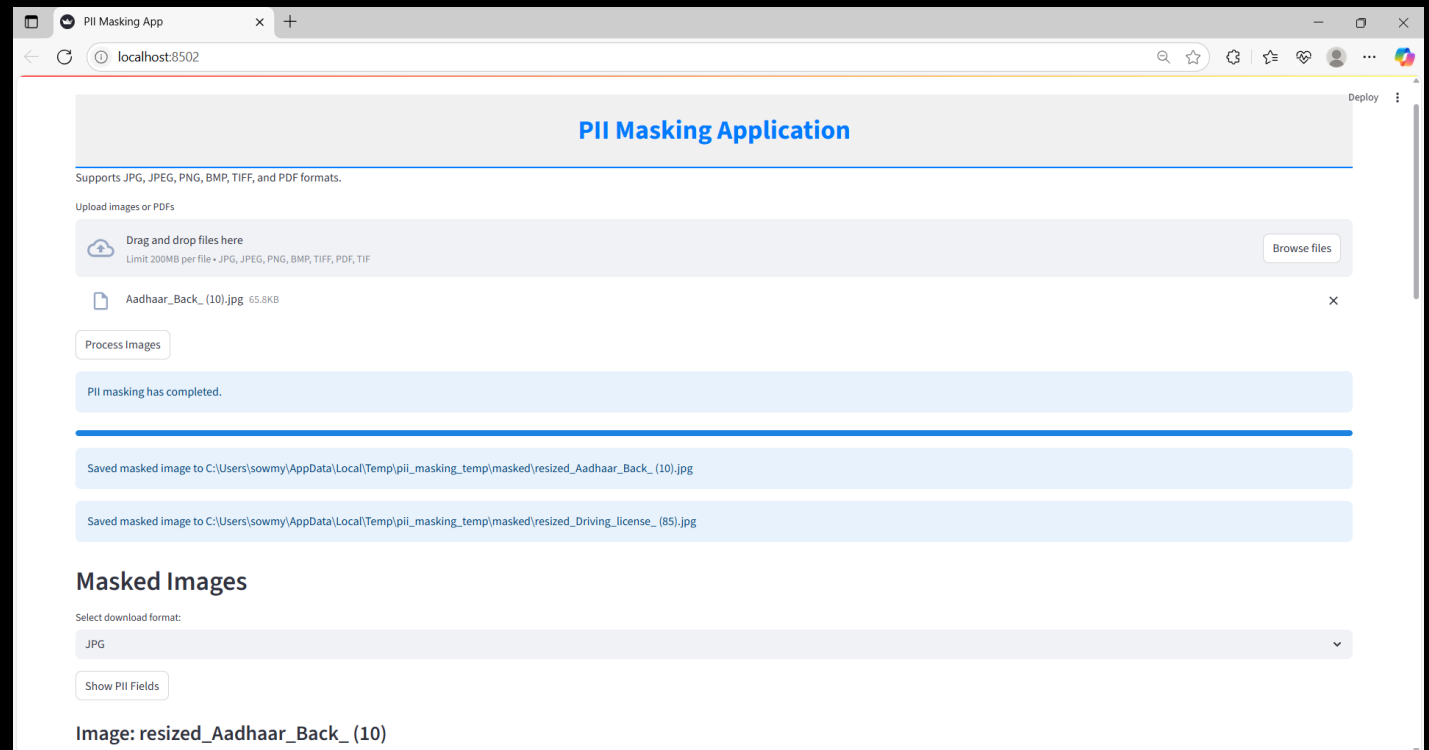
Process :

- Use the sensitive data coordinates identified by Gemini 1.5 Flash.
- Apply masking using OpenCV.
- Ensure non-sensitive content remains clear and readable.

Outcome : Images with sensitive data hidden, usable documents.

Streamlit App

- Built a simple web app using Streamlit where users can upload a document image. The app detects sensitive text automatically, masks it. The user preview the PII fields and download the safe version. It supports multiple file types.
- **How it works :**
 - Uploaded files are analyzed to detect PII and detected areas are automatically masked on the image
 - Users can preview the result before downloading
- **Output options :**
 - Download single file : Masked file can be saved as JPG, PNG, PDF
 - Download multiple files : All masked files can be downloaded as a single ZIP file



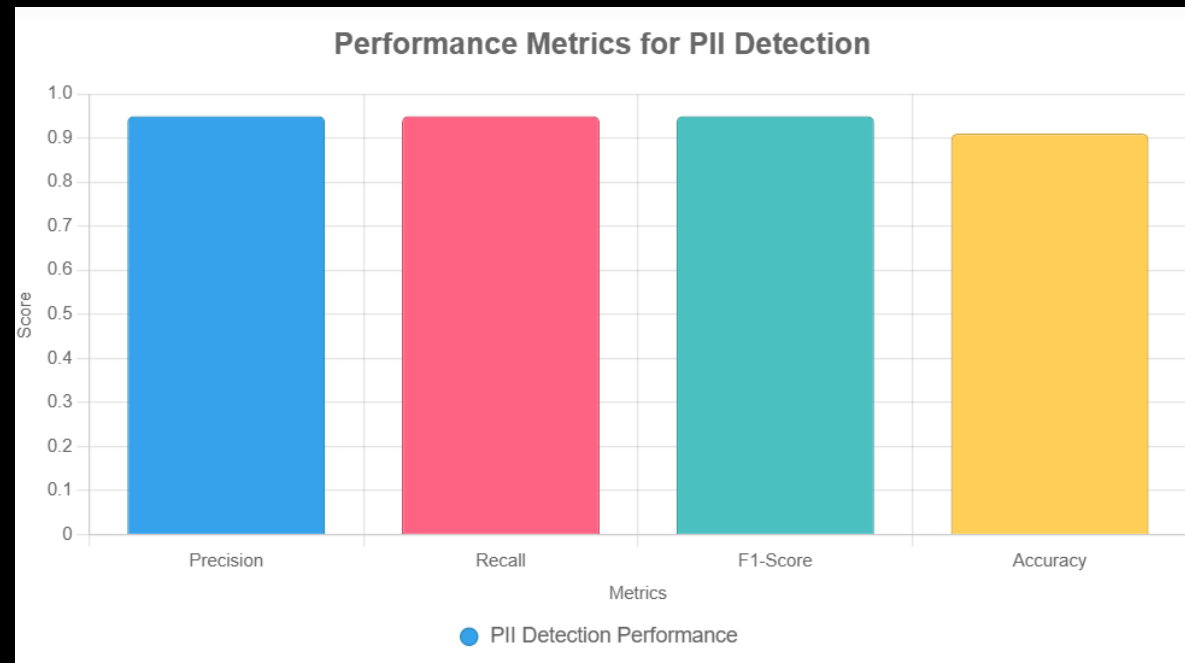
Results and Visualization



Overall PII Detection Results:

	Predicted : PII present	Predicted : No PII
Actual : PII present	TP = 984	FN = 51
Actual : No PII	FP = 52	TN = 57

- Coordinate Validity Rate: 0.27 (valid coordinates)
- Coordinate FP: 28
- Coordinate FN: 27
- The model shows strong text detection accuracy with a high F1-score and balanced precision/recall.
- Bounding box coordinate extraction (Low coordinate validity) needs improvement – possibly through better OCR alignment
- Overall, it's a high-performing detection system with potential for further refinement.



Challenges Faced

Challenge : ID card Embedded in Text with PII

- **Issue** : When an ID card is embedded in a document with surrounding text containing PII, YOLOV8 crops the ID card, excluding the surrounding text. This misses critical PII, risking privacy breaches.

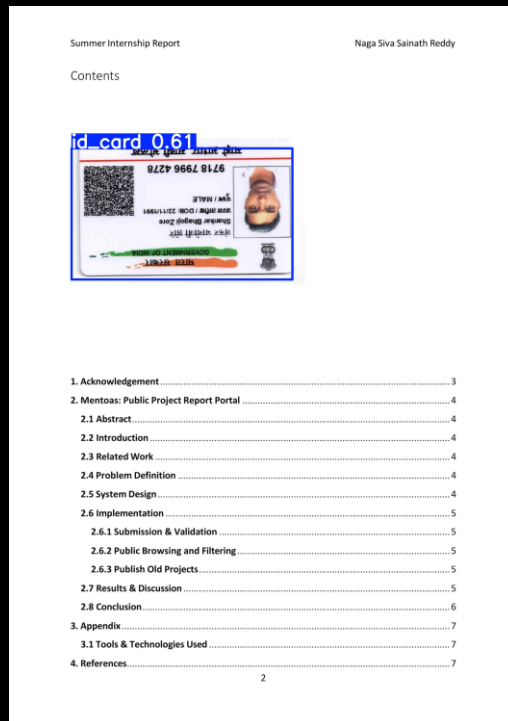


Solution : Selective Cropping with OCR

Use OCR to detect text around the ID card before cropping. If significant text is found outside the ID card's bounding box, process the entire image to capture all PII.

Steps :

- **YOLOV8 detection** : Identify ID card and their bounding boxes using YOLOv8.
- **OCR Analysis** : Apply PaddleOCR to locate text blocks in the original image
- **Text Checking** : Skip cropping if text exists outside the ID card's bounding box
- **Fallback** : Use original image if no ID card or significant text is detected



Challenges Faced

- **Alignment** : Fixing tilted images was tricky; sometimes even straight images got slightly rotated during processing.
- **OCR Extraction** : Didn't work well on poor-quality images; some text was not extracted.
- **Text Separation and Coordinate Accuracy** : PaddleOCR often gave one box for both label and value (e.g., "NAME: Shankar"); used keywords like "Name", "DOB" to split, but still missed a few cases.
- **YOLO Detection**: Found multiple ID cards in busy backgrounds; used confidence scores to select the best match.
- **Gemini** : Limited to 50 requests per day.

Future Scope



Multi – Language Support : Add support to detect PII in non-English languages like Hindi, Telugu, etc.








Better OCR Model : Use a more accurate OCR tool to improve text reading and coordinate extraction, especially on low-quality images.



Diverse Documents : Expand to handle **bank statements, credit/debit cards, hospital records**, and other types of documents.

Conclusion

-  Summary : Developed an automated application to detect and hide sensitive information in document images
-  Performance : Achieved high accuracy across varied document types
-  Privacy Protection : Hides the personal information while keeping non-sensitive data readable.
-  Security Impact : Improves privacy and reduces the risk of leaks and misuse of personal information.
-  Future Scope : This can be expanded to support multiple languages and more diverse document and image types in various fields where privacy is critical

Thank You