# CYCV000 Report

## 1 Dataset Choice and Reason

### 1.1 Dataset Selection

The primary dataset chosen for this project is the **“Egyptian National IDs”** dataset available on Kaggle, provided by user Yahya Ahmed. This dataset was selected after a thorough evaluation of available public resources.

### 1.2 Rationale for Selection

The choice of this specific dataset was due to three key reasons to align with the project’s objectives:

1. **Domain Specificity and Scale:** The dataset is the largest publicly available collection of Egyptian National ID cards, containing **1,931 annotated images**. Training YOLO model on a large volume data is the most effective way to ensure the model learns the specific features, layouts, colors, and text styles of the target documents. This significantly reduces the risk of the model failing to generalize on the final test cases.
2. **Annotation Quality and Task Alignment:** The annotations provided are high-quality bounding boxes drawn tightly around the entire ID card. This aligns perfectly with the first stage of the pipeline: **ID Card Detection**. The annotations were clean and required minimal preprocessing before being used for training the YOLO model.
3. **Superiority to Alternatives:** Other potential datasets, such as those found on Roboflow Universe, were considered. However, these alternatives were significantly smaller (typically 200–300 images) and often contained a mix of different document types. And also had unneeded annotations.

### 1.3 Dataset Split

The dataset was split into training, validation, and test sets using an 80/10/10 split:

| **Set** | **Number of Images** | **Percentage** |
|---|---|---|
| Training | 1,545 | 80% |
| Validation | 193 | 10% |
| Test | 193 | 10% |

# 2 Model Architecture

The system employs a two-stage architecture. The first stage uses a state-of-the-art deep learning model for robust ID card detection. The second stage consists of a deterministic pipeline of classical computer vision algorithms for alignment and line segmentation.

## 2.1 Stage 1: ID Card Detection (YOLOv11)

For the critical first step of accurately locating the ID card within an image, the **YOLOv11** model was selected. This choice was due to its balance of high accuracy, real-time inference speed, and enhanced performance on objects of varying scales.

The YOLOv11 architecture is composed of three primary components:

1. **Backbone:** This is the core feature extractor. It processes the input image through a series of convolutional layers (including efficient `C3K2` blocks) to generate feature maps at different scales. This hierarchical feature extraction is crucial for identifying objects of various sizes.

2. **Neck:** The neck of the model is responsible for fusing the feature maps generated by the backbone. It employs mechanisms like **SPFF (Spatial Pyramid Pooling Fast)** and a Path Aggregation Network (PAN) structure to combine rich semantic information from deeper layers with fine-grained spatial information from earlier layers. This fusion is vital for both accurate localization and classification. The inclusion of advanced attention mechanisms like **C2PSA** further refines these features, allowing the model to focus on the most salient regions of the image.

3. **Head:** The final stage of the model, the detection head, takes the fused feature maps from the neck and performs the final predictions. It outputs the bounding box coordinates and the confidence score for each detected ID card.

## 2.2 Stage 2: Alignment and Segmentation Pipeline (OpenCV)

After the detection of the ID card, a pipeline built with OpenCV processes the cropped image to extract individual lines of text. This pipeline is deterministic and does not require further training.

1. **Orientation Correction:**

   - **Fine Skew Detection:** The pipeline first detects minor rotational skew by finding the largest contour in the cropped image and calculating the angle of its minimum area rectangle (`cv2.minAreaRect`).

   - **Coarse Orientation:** It then uses template matching (`cv2.matchTemplate`) with a pre-saved image of the ID card's header to determine the correct coarse orientation ($0°$ + Fine Skew, $90°$ + Fine Skew, $180°$ + Fine Skew, or $270°$ + Fine Skew). The image is rotated to the angle with the highest matching score.

2. **Line Segmentation:**

   - The cleaned card image is converted to grayscale and binarized using **Otsu's Thresholding**, which automatically separates the text from the background.

- `cv2.findContours` is used to find the contours of all individual text elements (characters or character groups).
- These contours are grouped into lines based on their vertical proximity, effectively merging disparate characters into a single text line.
- Finally, a bounding box is drawn around each line group, and the corresponding image region is cropped and saved as the final output.

## 3 Loss Function Selection and Reason

The model training in this project involved fine-tuning a pre-trained **YOLOv11** model, rather than designing a new architecture from the ground up. Thus, the choice of loss function was inherent to the YOLOv11 framework.

The overall loss in a YOLO model is a composite of three separate components, each targeting a different aspect of the object detection task:

### 3.1 Bounding Box (Regression) Loss

- **Function:** YOLOv11 typically uses a variant of **CIoU (Complete Intersection over Union)** or a similar advanced IoU-based loss.
- **Reason:** The goal of this loss is to measure how well the predicted bounding box overlaps with the ground truth bounding box. Simple regression losses (like L1 or L2 distance) are not ideal as they do not directly correlate with the primary evaluation metric, IoU. CIoU is a superior choice because it accounts for three key geometric factors:
  1. **Overlap Area (IoU):** The primary measure of how much the boxes overlap.
  2. **Center Point Distance:** Penalizes the model if the center of the predicted box is far from the center of the ground truth box.
  3. **Aspect Ratio:** Penalizes the model if the shape (width vs. height) of the predicted box is different from the ground truth.
- **Conclusion:** By optimizing for CIoU, the model is trained to produce bounding boxes that are not only in the right location but are also the correct size and shape, which is critical for the subsequent cropping step in our pipeline.

### 3.2 Objectness (Confidence) Loss

- **Function:** This is typically a **Binary Cross-Entropy (BCE) with Logits Loss**.
- **Reason:** Every potential bounding box (anchor) in the output grid needs a score indicating its confidence that it contains an object (in this case, an ID card). This is a binary classification problem: does this box contain an object, or is it background? BCE is the standard and most effective loss function for this type of binary task.

### 3.3 Classification Loss

- **Function:** Also a **Binary Cross-Entropy (BCE) with Logits Loss**.
- **Reason:** For each box that is predicted to contain an object, this loss measures how well the model classifies that object. In this project, there is only one class ("egyptian_id"). While it may seem simple, this loss is still crucial for teaching the model to distinguish the target class from any other potential objects it might have learned to recognize in its pre-training.

Using these loss functions ensures the model is accurate in localization, confidence, and classification, making it a robust and well-founded choice for this detection task.

## 4 Performance Analysis and Evaluation Metrics

The system's performance was evaluated in two stages. The ID card detection model was assessed using rigorous quantitative metrics, while the end-to-end performance of the alignment and segmentation pipeline was validated through analysis on a diverse set of test images.

### 4.1 ID Card Detection Performance (Quantitative)

The YOLOv11 model was trained for 100 epochs with an early stopping mechanism to prevent overfitting, with training concluding at epoch 42. The model's performance was then benchmarked on the held-out test set, which consisted of 193 images containing 203 ID card instances.

The model achieved outstanding results, demonstrating its high accuracy and reliability:

| **Metric** | **Score** | **Description** |
|---|---|---|
| Precision | 0.98 | Of all the bounding boxes the model predicted, 98% were correct. |
| Recall | 0.988 | The model successfully found 98.8% of all actual ID cards present in the test images. |
| mAP@.50 | 0.994 | Near-perfect performance at the standard IoU threshold of 0.5, indicating excellent detection accuracy. |
| mAP@.50–.95 | 0.928 | A very strong average performance across stricter IoU thresholds, proving the model produces tight, precise bounding boxes. |

**Conclusion:** With a mAP50–95 score of **0.928**, the detection model serves as a highly reliable foundation for the rest of the pipeline. It can consistently and accurately locate ID cards, which is the critical first step for the entire system.

## 4.2 Full Pipeline Performance (Qualitative)

To evaluate the practical success of the entire system, a qualitative analysis was performed. A test suite of 8 real-world images was used. This set was specifically chosen to include a variety of difficult conditions:

- Different rotation angles (minor skew, 90 degrees, and 180 degrees).
- Images containing multiple ID cards.
- Varying lighting and background conditions.

The system was **successful in all 8 test cases**.

- **Alignment:** The pipeline correctly identified and corrected the orientation for every card, producing straight, upright images.
- **Segmentation:** After alignment, the system successfully isolated and extracted the individual text lines from each card, meeting the core objective of the assignment.

**Note:** You can find the test cases results in `demo.ipynb` in the repo.

## 4.3 Test case

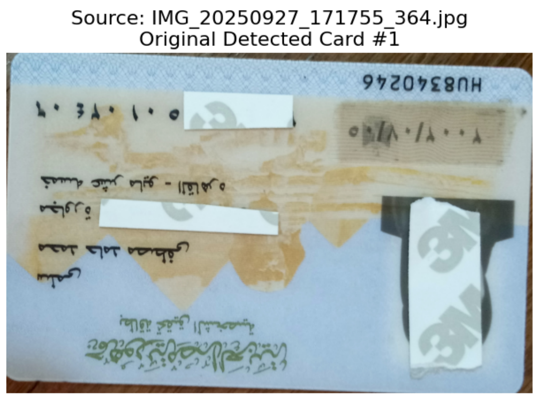
Source: IMG_20250927_171755_364.jpg
Original Detected Card #1

Figure 1: Input Image

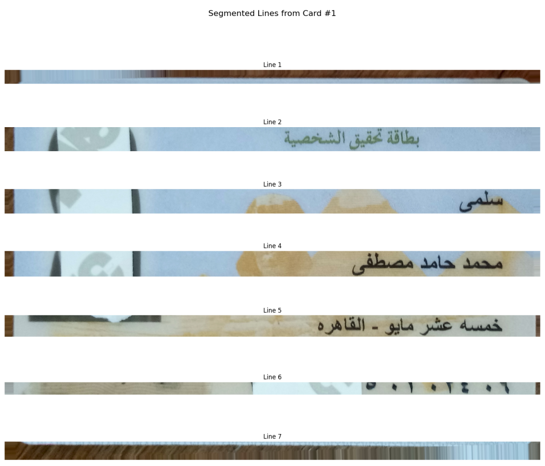

Figure 2: Output Image

This qualitative validation confirms that the components of the pipeline work together effectively to transform a raw input image into a set of clean, usable line segments.

## 5 Limitations of the System

While the system performs effectively under a wide range of conditions, it is important to acknowledge its limitations. The primary limitation stems from the sequential nature of the pipeline, where the performance of later stages is highly dependent on the precision of the initial detection stage.

### 5.1 Sensitivity to Bounding Box Precision

- **The Problem:** The system is sensitive to the tightness of the bounding box generated by the YOLOv11 model. If the YOLO model produces a bounding box that is significantly larger than the actual ID card, it can cause background elements to be included in the cropped image. These background elements can be incorrectly processed and segmented as “faulty” lines.

### 5.2 Handling of Severe Perspective Distortion

- **The Problem:** The current alignment model is excellent at correcting rotational skew on a 2D plane. However, it is not designed to handle severe perspective distortions, which occur when a photo of an ID card is taken from a sharp angle. This distortion can cause some lines of text to appear compressed or skewed in a non-linear way, potentially leading to failed segmentation.

### 5.3 Template Dependency for Orientation

- **The Problem:** The coarse orientation correction relies on template matching with a specific header image (`header_template.png`). While effective for the current ID card version, this method would fail if the layout of the ID card changes in the future (e.g., a new design is issued).