Egyptian License Plate Classification System

Vo Ngoc Tram Anh

October 27, 2025

Contents

1 Introduction

1.1 Egyptian License Plate System Overview

The Egyptian vehicle registration system employs color-coded license plates to distinguish vehicle categories. Each plate features the word "Egypt" in both English and Arabic, along with a colored rectangle indicating the vehicle type. Registration codes combine Arabic numerals and letters, which correspond to different governorates [1]. The main categories include Private (Blue), Taxi (Orange), Commercial (Red), Public Transport (Grey), Diplomatic (Green), and Tourist/Temporary (Yellow). The overall structure of the Egyptian license plate is illustrated in Fig. 1.

Figure 1: Egyptian License Plate Structure [1]

1.2 Classification Challenge Description

The objective of this task is to develop an automated system capable of classifying Egyptian vehicle license plates into six official categories based on their visual appearance. Each plate type corresponds to a distinct vehicle registration class, identified primarily through its background color: *Blue (Private)*, *Orange (Taxi)*, *Red (Commercial)*, *Grey (Public Transport)*, *Green (Diplomatic)*, and *Yellow (Tourist/Temporary)*.

Given cropped images of license plates extracted from real Egyptian traffic videos, the model must accurately predict the plate type by learning discriminative visual cues, especially color. The classification problem is formulated as a six-class supervised learning task using deep learning architectures, with key challenges arising from lighting variations, class imbalance, and environmental noise in real-world conditions.

In this work, I assume that the input images are tightly cropped around the license plate region to minimize background interference while preserving the essential color and shape characteristics of the plate. This assumption is intended to enhance the model's focus on relevant visual features and improve overall classification performance.

1.3 Real-World Applications and Significance

Color-based license plate classification serves as a vital component in intelligent transportation systems (ITS), particularly in countries like Egypt, where plate color directly encodes vehicle type. Accurate classification of plate colors enables automated systems to infer vehicle categories without relying on character recognition, providing faster and more robust decision-making in various real-world contexts.

Practical applications include vehicle-type-based access control, parking management, traffic analytics, toll collection, and law enforcement. For example, distinguishing between private, commercial, and public transport vehicles allows for more efficient traffic regulation and infrastructure planning. Moreover, identifying diplomatic or temporary vehicles supports enhanced security monitoring and exception handling in restricted zones.

From a research standpoint, developing a reliable Egyptian plate color classification model contributes to advancing computer vision systems for region-specific datasets. It also establishes a foundation for future work on end-to-end vehicle classification and traffic intelligence using visual cues beyond textual recognition.

2 Data Analysis

2.1 Video Source Analysis and Frame Extraction Methodology

2.1.1 Video Source Analysis

The dataset was constructed from 24 publicly available YouTube videos featuring Egyptian vehicles captured in real-world environments. These videos, listed below, were recorded by various independent creators and thus exhibit a wide range of visual characteristics, including differences in lighting, resolution, and shooting perspectives.

https://www.youtube.com/watch?v=dGcEfxSG4eM

https://www.youtube.com/watch?v=6v8kbz8ZxlM

https://www.youtube.com/watch?v=vIXW23oYxlA

https://www.youtube.com/watch?v=z3KQSdc3vQg

https://www.youtube.com/watch?v=uYionr1X-Jw

https://www.youtube.com/watch?v=OqfWD2XGSZ0

https://www.youtube.com/watch?v=n3Bv5JmGsYQ

https://www.youtube.com/watch?v=hvS5RC2P664

https://www.youtube.com/watch?v=1RZ2dmkV2Rg

https://www.youtube.com/watch?v=r31DFG5idLc

https://www.youtube.com/watch?v=00xbFDbSZy0

https://www.youtube.com/watch?v=tY2oeBh2LZc

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

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

https://www.youtube.com/watch?v=cv7w_dJq8pg

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

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

https://www.youtube.com/watch?v=VaTQXfKE6Pw

https://www.youtube.com/watch?v=25DlLwARtlA

https://www.youtube.com/watch?v=SuuFB8thgFo

https://www.youtube.com/watch?v=dlbH9d64xGo

https://www.youtube.com/watch?v=xExj4__wBKc

https://www.youtube.com/watch?v=xYYQ5z9TyNs

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

These videos were recorded in diverse urban contexts - such as administrative districts, touristic zones, and main roads - resulting in a naturally varied distribution of vehicle types. For instance, touristic areas typically contain more taxis, while administrative regions more frequently include diplomatic cars. Furthermore, the footage spans different times of day (morning, noon, evening, and night), leading to significant variation in illumination and environmental appearance. This diversity in spatial, temporal, and contextual factors enhances the dataset's representativeness for real-world classification tasks.

2.1.2 Frame Extraction Methodology

The initial frame extraction approach relied on OpenCV's `cv2.VideoCapture()` to automatically sample frames at fixed intervals. However, this method often resulted in redundant or unrepresentative frames and frequently missed rare plate types (e.g., yellow or green), which might only appear briefly in the video.

To ensure higher data quality and completeness, a fully manual frame extraction approach was adopted. Each video was watched in its entirety, and frames were captured manually whenever a license plate appeared clearly in view.

- It enabled comprehensive coverage, ensuring that even short, rare occurrences of certain plate categories were not overlooked.
- It allowed real-time judgment and filtering, letting the operator skip frames with excessive motion blur, occlusion, or confusing reflections.
- It supported context-aware labeling, since viewing the full video helped confirm ambiguous cases - for instance, when an orange taxi plate appeared reddish under specific lighting.

2.2 Class Distribution and Data Quality Assessment

A comprehensive evaluation of the dataset was conducted to analyze class balance, verify data integrity, and design an appropriate augmentation pipeline. All preprocessing, splitting, and augmentation procedures were implemented in Python using the libraries Pandas, OpenCV, Albumentations, and Matplotlib.

The raw dataset was organized under the root directory `Data/`, with each subfolder corresponding to one of six target classes, as summarized in the following table:

Table 1: Folder structure and class mapping of the raw dataset

Folder Name	Category	Plate Color
`1_Private_Blue`	Private	Blue
`2_Taxi_Orange`	Taxi	Orange
`3_Commercial_Red`	Commercial	Red
`4_PublicTransport_Grey`	Public Transport	Grey
`5_Diplomats_Green`	Diplomats	Green
`6_TouristTemporary_Yellow`	Tourist / Temporary	Yellow

A Python script iterated through each folder and counted valid image files (`.jpg`, `.jpeg`, `.png`). The resulting class distribution was compiled into a DataFrame and visualized using a **bar chart**. The analysis revealed a severe class imbalance, with *Private (Blue)* plates being the dominant class, while *Diplomats (Green)*, *Public Transport (Grey)* and *Tourist/Temporary (Yellow)* were notably underrepresented.

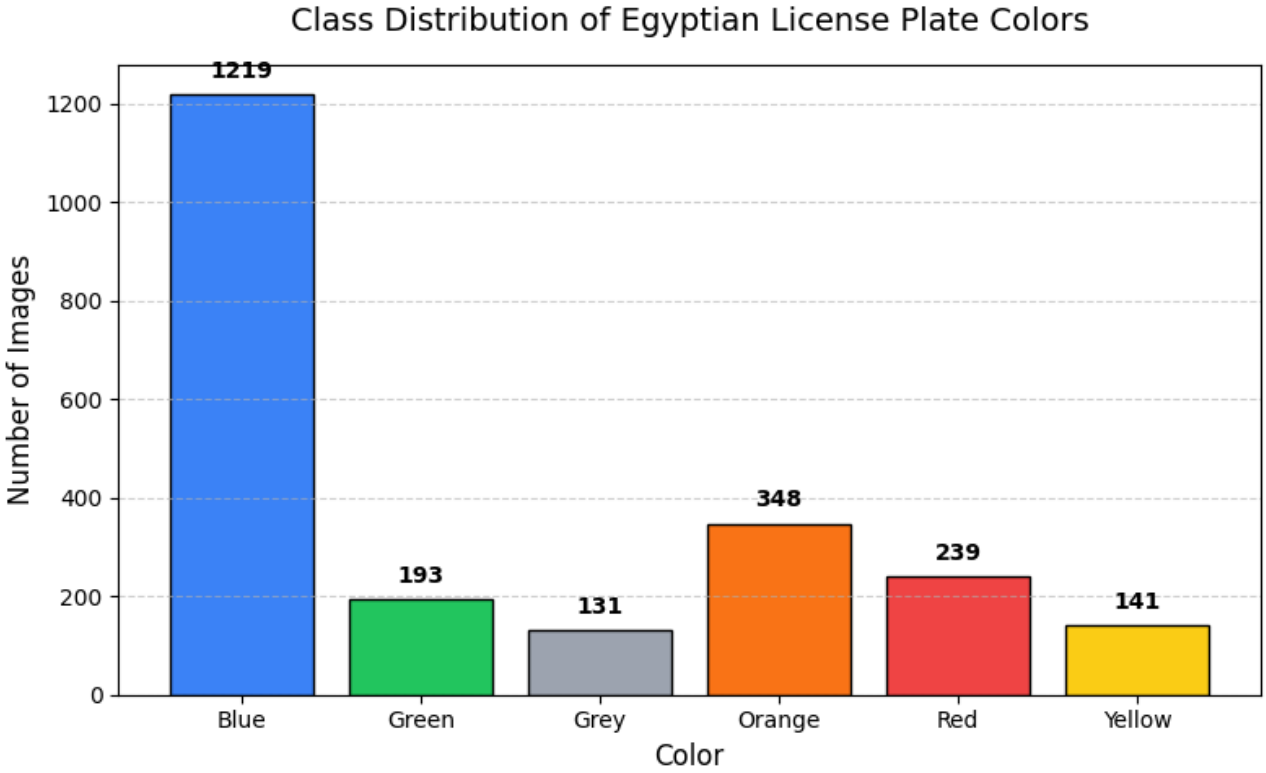


Figure 2: Class distribution of the raw dataset before train-test split and augmentation

2.2.1 Train-Test Split Strategy

Given the limited number of samples in some minority classes, a stratified but conservative split was adopted: Each class contributed **35 randomly selected images** to the test set, while the remaining samples were assigned to the training set. This ensured a balanced test set without excessively reducing the training data for rare classes (e.g., *Green*, *Yellow*).

The splitting process was implemented with controlled randomness using a fixed seed (`SEED = 42`) to ensure reproducibility.

2.2.2 Data Augmentation Methodology

To address class imbalance and improve model generalization, a targeted augmentation strategy was designed using the *Albumentations* library. After experimenting with multiple transformations (e.g., zoom, blur, hue shift, rotation), the final augmentation pipeline prioritized **color preservation**, since color is the core discriminative feature in license plate classification.

The final augmentation transformations were:

Table 2: Augmentation operations and configuration parameters

Transformation	**Range**	**Probability (p)**	**Purpose**
`A.Rotate`	$\pm 7°$	0.5	Simulate camera angle variation
`A.HorizontalFlip`	—	0.3	Increase spatial diversity
`A.RandomScale`	$\pm 10\%$	0.3	Handle distance variability
`A.RandomBrightnessContrast`	± 0.02	0.3	Model mild lighting variation

Each minority class (all except Blue) was augmented to reach a **target of 600 training images per class**. During augmentation, a random base image was selected and transformed probabilistically, producing diverse but color-faithful variations. This process was designed to:

- Improve intra-class diversity without distorting true plate colors.
- Reduce overfitting risk by introducing controlled randomization.
- Bring the dataset closer to class balance.

The final distribution after augmentation is shown in the following figures:

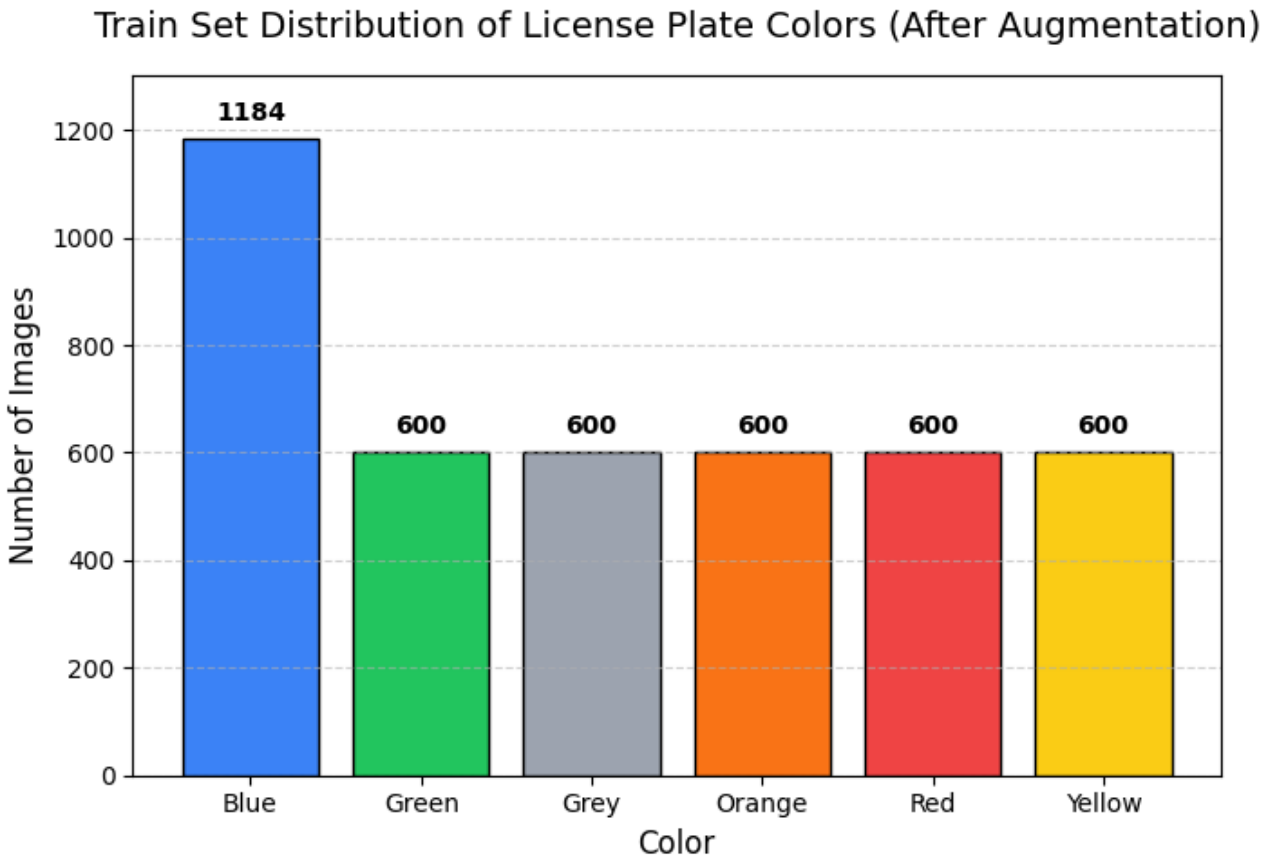


Figure 3: Class distribution of training set after augmentation

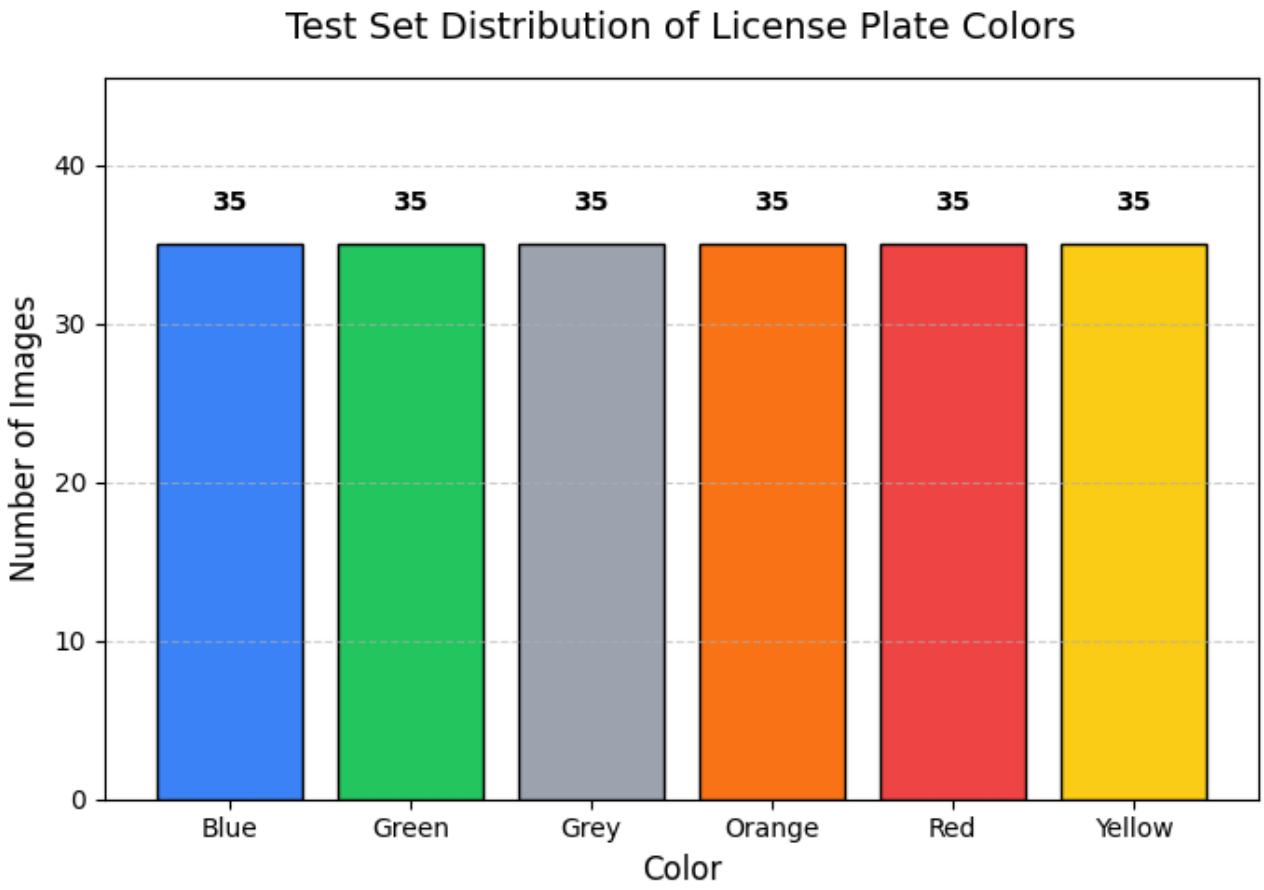


Figure 4: Class distribution of testing set

2.2.3 Visual Validation and Quality Control

To ensure augmentation integrity, 20 randomly selected samples per class were displayed for visual verification. Inspection confirmed that class-defining color characteristics were preserved while achieving greater variety in brightness, scale, and orientation.

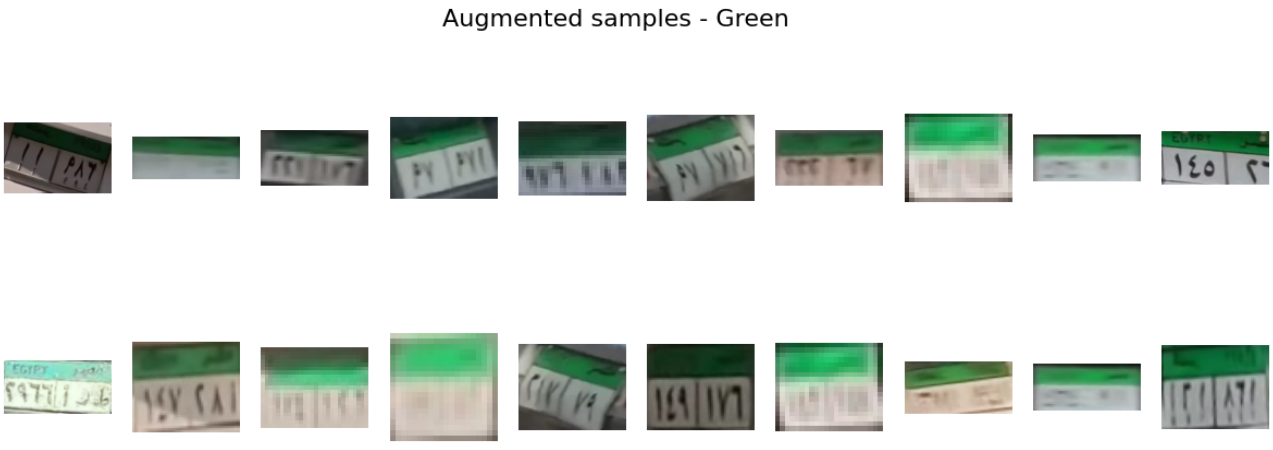


Figure 5: Representative augmented samples showing preserved color characteristics

Additionally, a full manual quality control (QC) was performed:

1. **Visual validation:** Ensured that the license plate region was clearly visible.

2. **Label verification:** Removed any mislabeled or ambiguous samples.

3. **Image quality check:** Discarded low-quality or overexposed images.

4. **Color fidelity assurance:** Verified that augmentation did not shift plate hues across categories.

This manual review was critical, as the dataset originated from multiple YouTube videos with heterogeneous environments and camera qualities. The resulting dataset achieved consistent color representation and strong inter-class separability, forming a robust foundation for model training and evaluation.

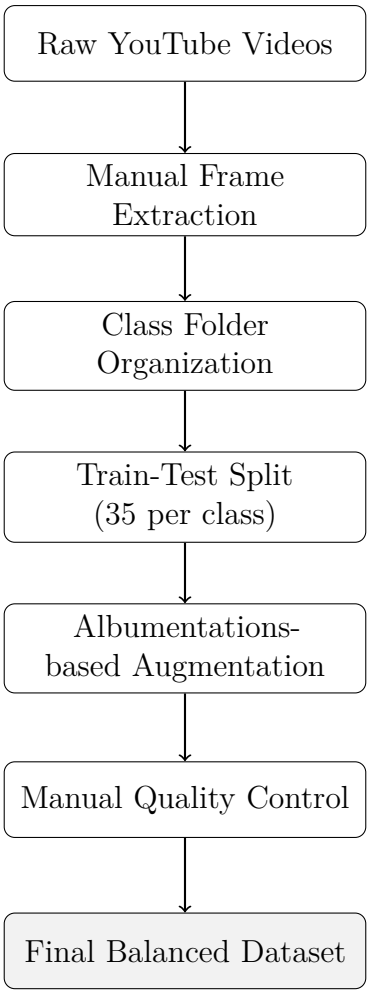


Figure 6: Overview of the dataset preparation and augmentation pipeline

2.3 Challenges Specific to Egyptian Traffic Conditions

Classifying Egyptian license plates is a fine-grained visual task due to subtle color and format differences, further complicated by lighting variations, occlusions, and class imbalance. During data collection, several key challenges were observed, as illustrated in the following examples:

- **Partial occlusion:** In some cases, license plates are partially obscured by surrounding objects such as other vehicles, pedestrians, or vehicle components. These occlusions reduce the visible area of the plate and may remove critical color or texture cues needed for accurate classification.

Figure 7: Occlusion caused by vehicle component

- **Lighting and environmental sensitivity:** The RGB color values and visual appearance of plates are affected by both illumination and environmental factors. Variations in brightness, glare, and shadow, as well as dust, surface wear, plate aging, or differences in camera/video quality, can cause the same plate to appear significantly different across scenes. These changes often degrade color clarity and complicate automated detection.

Figure 8: The same plate color under different environmental and lighting conditions

- **Color similarity:** Some categories, particularly orange (Taxi) and red (Commercial) plates, share close hue and saturation ranges. Under certain lighting conditions, their distinction becomes visually ambiguous, even for human observers.

Figure 9: Color similarity between visually close categories

- **Perspective distortion:** Plates captured from oblique angles or at varying distances exhibit geometric distortions that alter perceived proportions and color balance. This variation complicates both localization and classification, especially when the plate occupies a small or skewed region within the image.

Figure 10: Perspective distortion across different camera viewpoints

- **Data imbalance:** The dataset distribution is dominated by blue (Private) plates, while other classes are significantly underrepresented. This imbalance was analyzed in "Initial Class Distribution Analysis".

These challenges highlight the need for robust preprocessing and data augmentation techniques to ensure reliable model generalization and stable color representation across varying environmental and lighting conditions.

3 Methodology

3.1 Model Architecture Rationale

In this project, I evaluated approximately 10 different deep learning architectures, focusing on performance metrics such as accuracy, precision, recall, F1-score, and training efficiency (including time and computational resources). From these experiments, I selected the top three models based on their overall performance on the test set: Simple CNN, EfficientNet-B0, and MobileNetV2, ranked in descending order of effectiveness.

3.1.1 Simple CNN

Initially, I approached the problem using deeper architectures such as ResNet-50, followed by ResNet-34 and ResNet-18. However, these models exhibited overfitting, likely due to the small input image size of 48x48 pixels, which is significantly smaller than the typical 224x224 input size for ResNet models trained on ImageNet. This mismatch can lead to inefficient feature extraction and increased susceptibility to overfitting on my dataset.

Consequently, I shifted my approach to simpler CNN architectures. After testing several variants, the selected Simple CNN provided the best performance, striking a balance between underfitting observed in even shallower models and the overfitting in more complex ones.

The Simple CNN was custom-designed to provide a balance between expressive capacity and training efficiency. It consists of three convolutional blocks, each followed by batch normalization, ReLU activation, and max pooling layers. This architecture allows hierarchical extraction of spatial features while mitigating overfitting through dropout regularization. The fully connected classifier maps the flattened feature representation to six output classes corresponding to different license plate color categories.

Table 3: CNN Architecture Summary

Layer (type)	**Output Shape**	**Param #**
conv2d (Conv2D)	(None, 48, 48, 32)	896
batch_normalization	(None, 48, 48, 32)	128
activation (Activation)	(None, 48, 48, 32)	0
max_pooling2d (MaxPooling2D)	(None, 24, 24, 32)	0
conv2d_1 (Conv2D)	(None, 24, 24, 64)	18,496
batch_normalization_1	(None, 24, 24, 64)	256
activation_1 (Activation)	(None, 24, 24, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 12, 12, 64)	0
conv2d_2 (Conv2D)	(None, 12, 12, 128)	73,856
batch_normalization_2	(None, 12, 12, 128)	512
activation_2 (Activation)	(None, 12, 12, 128)	0
max_pooling2d_2 (MaxPooling2D)	(None, 6, 6, 128)	0
flatten (Flatten)	(None, 4608)	0
dense (Dense)	(None, 256)	1,179,904
activation_3 (Activation)	(None, 256)	0
dropout (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 6)	1,542

3.1.2 EfficientNetB0

EfficientNetB0 employs a compound scaling strategy that uniformly scales network depth, width, and input resolution. Its architecture, based on the Mobile Inverted Bottleneck Convolution (MBConv) and squeeze-and-excitation optimization, achieves superior parameter efficiency and accuracy [2]. As a result, it serves as a strong baseline for balanced trade-offs between accuracy and training time.

3.1.3 MobileV2.

MobileNetV2 was selected as a lightweight yet powerful transfer learning baseline. Its inverted residual and depthwise separable convolution design significantly reduce computational cost while maintaining strong representational power [3]. This makes it particularly suitable for deployment in resource-constrained environments.

3.2 Training Strategy

To ensure a rigorous and unbiased evaluation, a 5-fold stratified cross-validation strategy was implemented. This approach divides the dataset into five equally sized folds, maintaining proportional class distribution across all folds. In each iteration, four folds were used for training and one for validation, ensuring that every sample was validated exactly once. The model achieving the lowest validation loss in each fold was saved as a checkpoint.

All experiments were conducted on the same pipeline to maintain consistency. The training process utilized the Adam optimizer with a learning rate of 1×10^{-3} and a categorical cross-entropy loss function. Each model was trained for a maximum of 50 epochs with early stopping applied when the validation loss did not improve for 5 consecutive epochs. The batch size was set to 16, and input normalization was performed with a mean and standard deviation of $[0.5, 0.5, 0.5]$.

After completing the cross-validation, the best model from each fold was evaluated on the held-out test set using classification reports and confusion matrices. This allowed me to select the overall best-performing model across all folds based on its test set performance. In addition, qualitative analysis of misclassified samples was conducted to better understand the model's weaknesses and potential areas for improvement.

3.3 Evaluation Metrics Selection

In this project, I used two main categories of evaluation metrics to assess both the performance and efficiency of the proposed models: **classification metrics** and **training efficiency metrics**. This dual perspective ensured that the selected model was not only accurate but also computationally practical for real-world deployment scenarios.

3.3.1 Classification Metrics

At the beginning of my experiments, I primarily focused on evaluating the models using classification-based metrics. The following metrics were employed to measure the predictive capability of each architecture across all classes:

- **Accuracy:** Represents the overall proportion of correctly classified samples among all predictions. While intuitive, accuracy alone may not fully capture class-level performance in multi-class classification.
- **Precision:** Measures the proportion of correct positive predictions out of all predicted positives for each class. High precision indicates a low false-positive rate.
- **Recall:** Indicates the proportion of actual positives that are correctly identified. High recall corresponds to a low false-negative rate.
- **F1-Score:** The harmonic mean of Precision and Recall, providing a balanced evaluation between the two metrics, especially useful in cases of class imbalance.

I used the **classification_report** function from *scikit-learn* to compute these metrics, which allowed both per-class and averaged (macro and weighted) performance analysis. Additionally, I plotted the **confusion matrix** to visualize misclassifications and better understand which categories were most frequently confused with each other (e.g., *Orange* vs. *Red* plates). This analysis provided valuable insights into the strengths and weaknesses of each model in handling visual similarities across classes.

3.3.2 Training Efficiency Metrics

While classification metrics provided an in-depth evaluation of prediction quality, they did not reflect the computational efficiency of the models. As I began comparing multiple CNN architectures, I realized that some models achieved slightly better accuracy at the cost of significantly longer training times or slower convergence. Therefore, I decided to include additional metrics to capture the efficiency aspect of model training:

- **Training Time per Fold:** The total duration required to train each model within a 5-Fold Cross Validation setting. This metric helped me estimate the computational cost and scalability of different architectures.
- **Convergence Speed:** The rate at which the training and validation losses stabilized. Faster convergence with lower validation loss was considered an indicator of better training efficiency.

By analyzing both predictive performance and training efficiency, I was able to select the model that offered the best trade-off between accuracy and computational cost. Initially, I relied only on classification reports and confusion matrices, but as the number of architectures grew, I realized that incorporating training time as a secondary criterion allowed for a fairer and more realistic comparison. This holistic evaluation ultimately guided my decision in selecting the most suitable model for deployment.

4 Results and Discussion

4.1 Quantitative Performance Analysis

Table 4 summarizes the quantitative results for all three architectures across 5-Fold cross-validation. Each model includes both the averaged performance across folds and the best-performing fold, highlighting stability and peak performance.

Table 4: Average and best-fold quantitative performance of three models.

Model	Fold Type	Accuracy	Precision	Recall	F1-score	Training Time (min/fold)
SimpleCNN	Average	0.99	0.99	0.99	0.99	1.37
	Best Fold	1.00	1.00	1.00	1.00	1.72
EfficientNet-B0	Average	0.98	0.98	0.98	0.98	2.95
	Best Fold	0.99	0.99	0.99	0.99	3.05
MobileNetV2	Average	0.95	0.95	0.95	0.95	2.73
	Best Fold	0.97	0.97	0.97	0.97	4.33

From Table 4, the Simple CNN consistently outperformed both EfficientNet-B0 and MobileNetV2 across all evaluated metrics. Its superior accuracy, precision, recall, and F1-score - achieving nearly perfect results in both average and best-fold evaluations - demonstrate that a carefully designed lightweight architecture can be more effective than more complex pre-trained networks when the dataset and input resolution are limited. In addition, the Simple CNN achieved this level of performance with the shortest average training time per fold (1.37 minutes), highlighting its computational efficiency and suitability for real-time or resource-constrained environments.

EfficientNet-B0 followed closely, maintaining stable performance across folds with slightly longer training times. This suggests that while its compound scaling approach provides robustness, it may not offer significant gains over a well-tuned custom CNN in small-scale, domain-specific tasks. MobileNetV2, although efficient by design, exhibited relatively lower performance and longer training durations, indicating that its depthwise separable convolutions might be less effective for smaller input resolutions and simpler datasets.

Overall, the quantitative analysis confirms that the Simple CNN not only generalizes well but also provides the best trade-off between predictive accuracy and training efficiency. However, despite its strong quantitative results, qualitative inspection of misclassified samples is necessary to understand potential weaknesses in class boundary learning - particularly among visually similar license plate colors. The following section presents a detailed **error analysis** to explore these failure cases and provide insights for further model refinement.

4.2 Error Analysis and Failure Cases

In this section, I selected the Simple CNN for detailed error analysis due to its superior quantitative performance (see Table 4). Below I present per-fold misclassification summaries, visualisation placeholders (misclassified samples and per-fold confusion matrices), pattern analysis, comparison with the other architectures, and concrete recommendations to reduce the observed errors.

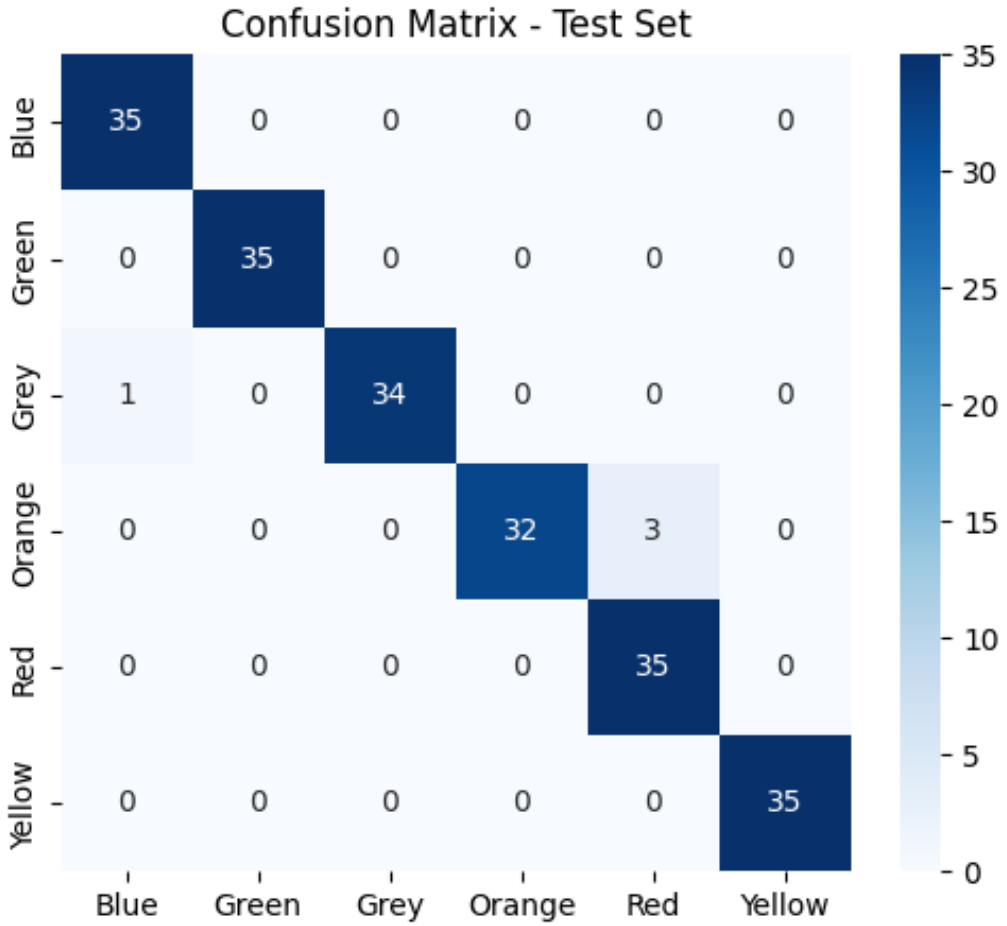
Table 5 summarises the misclassified samples observed for the Simple CNN across the 5 folds. Each reported case includes the true label, the predicted label, and the model's softmax confidence for the predicted class.

Table 5: Per-fold misclassification summary for Simple CNN

Fold	**# errors**	**Cases (true → pred)**	**Confidence(s)**
Fold 1	2 / 210	grey → blue; orange → red	0.88; 0.92
Fold 2	4 / 210	grey → blue; orange → red (3 cases)	0.72; 0.99; 0.79; 0.70
Fold 3	0 / 210	— perfect on this fold	—
Fold 4	3 / 210	grey → blue (2 cases); orange → red	0.61; 0.77; 0.91
Fold 5	2 / 210	yellow → grey; yellow → orange	0.98; 0.69

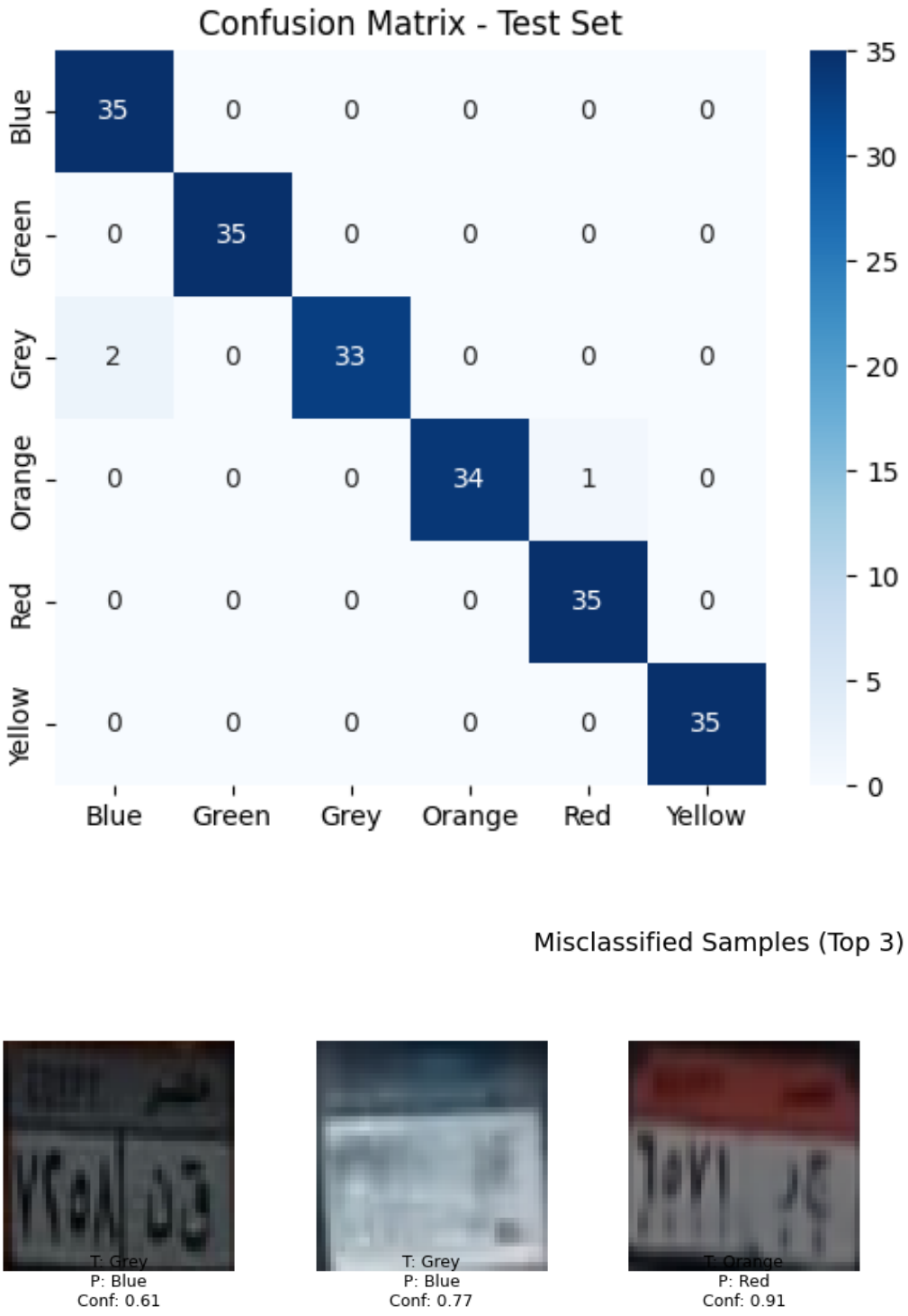
Fold 1 (2 / 210 errors). Visual inspection of these two misclassified samples (*grey→blue* and *orange→red*) reveals that both are affected by illumination artifacts. Even to the human eye, these examples appear ambiguous, supporting the interpretation that the model's high-confidence mistakes (0.88, 0.92) stem from difficult lighting conditions rather than representation failure.

Fold 2 (4 / 210 errors). Four errors occurred in this fold - again dominated by *grey→blue* and multiple *orange→red* cases. The highest-confidence mistake (0.99) suggests a consistent color confusion pattern rather than random noise. When visualized, these samples show similar ambient-light conditions: orange plates appear slightly desaturated and reddish under shadow. The remaining lower-confidence errors (0.79, 0.70) likely result from weak chromatic cues after resizing to 48×48, making the two adjacent hues less separable in feature space.

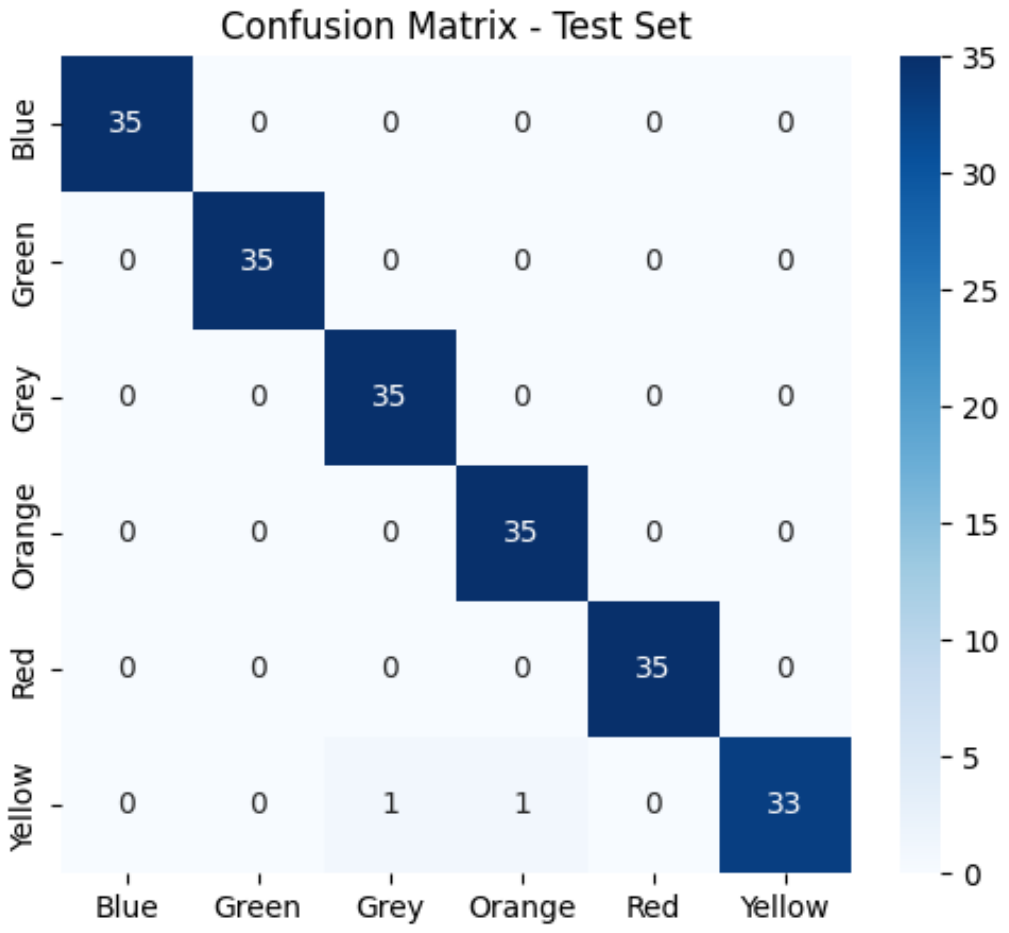


Fold 3 (0 / 210 errors). This fold achieved perfect classification accuracy, showing that the model can form a clean and robust decision boundary when the validation subset lacks extreme lighting or color overlap. This may also indicate that the data distribution across folds is not entirely uniform - i.e., some lighting or hue-specific edge cases cluster in other folds.

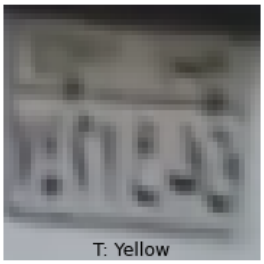
Fold 4 (3 / 210 errors). Two *grey→blue* and one *orange→red* cases reappear, confirming a persistent confusion pattern across folds. The lower-confidence grey→blue samples (0.61, 0.77) suggest borderline color regions near the classification threshold, which could benefit from targeted augmentation such as color jittering or histogram equalization. Meanwhile, the orange→red confusion again shows high certainty (0.91), reflecting the strong color adjacency, which was claimed in section "Challenges Specific to Egyptian Traffic Conditions".



Fold 5 (2 / 210 errors). This fold uniquely features errors involving the *yellow* class - one misclassified as *grey* (0.98) and another as *orange* (0.69). Visual inspection suggests that intense sunlight reflections and white balance clipping washed out the yellow chroma, making these samples closer to pale grey or orange. The 0.98 confidence indicates that illumination variability can still mislead the network decisively, even after extensive training.



Misclassified Samples (Top 2)



Cross-fold pattern and comparison. Across all folds, the same recurring confusion pairs emerge: (*grey*↔*blue*) and (*orange*↔*red*), occasionally (*yellow*↔*grey/orange*). These correspond exactly to plate colors with overlapping hue spectra under variable sunlight, confirming that illumination and desaturation are dominant error sources.

Compared with MobileNet and EfficientNet, the Simple CNN exhibits more **localized and systematic errors** - its mistakes are clustered in a few predictable color pairs rather than scattered across many classes. This suggests that while MobileNet and EfficientNet generalize more broadly, the CNN's simpler decision space may be easier to improve through **targeted augmentation** (e.g., color jitter, hue normalization) and balanced color sampling. In other words, the CNN's confusion pattern is more structured and thus potentially easier to correct in subsequent training iterations.

4.3 Comparison with Baseline Approaches

The comparative analysis between the proposed Simple CNN and the two transfer learning baselines (EfficientNet-B0 and MobileNetV2) reveals clear distinctions in both quantitative performance and qualitative behavior.

Quantitative comparison. As shown in Table 4, the Simple CNN achieved the highest overall performance across all evaluated metrics - accuracy, precision, recall, and F1-score - surpassing both baselines by a notable margin while also maintaining the shortest training time per fold. This demonstrates that, despite its shallower depth, the custom CNN was better suited to the low-resolution (48×48) input domain than the deeper pretrained networks originally optimized for 224×224 ImageNet images. EfficientNet-B0 followed closely with stable results but at more than twice the computational cost, while MobileNetV2 showed the lowest accuracy and longest convergence time, suggesting that its depthwise separable convolutions are less effective for this small-scale color classification task.

Qualitative and error-based comparison. Error inspection further emphasizes these trends. The Simple CNN's misclassifications were **highly localized** - primarily involving a few recurrent color pairs such as *grey*↔*blue* and *orange*↔*red*. In contrast, both EfficientNet-B0 and MobileNetV2 exhibited a broader distribution of confusion across multiple classes, implying that their learned feature spaces were less specialized for the dataset's specific hue distributions. This structured error behavior in the Simple CNN suggests that its failure modes are **systematic rather than random**, making them easier to identify and address through targeted color augmentation or illumination normalization.

Architectural interpretation. From an architectural standpoint, the Simple CNN's effectiveness can be attributed to its balanced depth and inductive bias toward local texture and color features. Larger pretrained networks, though more expressive, may suffer from feature redundancy and overparameterization relative to the dataset size, leading to slower convergence and occasional overfitting. The Simple CNN, by contrast, effectively captures low-level chromatic cues while maintaining strong generalization due to regularization mechanisms such as batch normalization and dropout.

Summary. In summary, while EfficientNet-B0 offers competitive accuracy with higher computational demand, and MobileNetV2 provides lightweight deployment capabilities with slightly lower accuracy, the Simple CNN delivers the best trade-off between performance, interpretability, and efficiency. Its limited yet consistent error distribution indicates that further improvements - particularly in illumination-aware augmentation - could yield near-perfect classification under real-world conditions.

5 Limitations and Future Work

5.1 Limitations and Improvement Opportunities

Despite the strong performance of the proposed Simple CNN, several limitations were observed, highlighting areas for further improvement:

Limited illumination robustness. The error analysis indicated that most misclassifications occurred under challenging lighting conditions, including shadows, glare, or overexposed regions.

High-confidence errors, such as *yellow→grey* or *orange→red*, demonstrate that even a well-trained network can be decisively misled by illumination variability. This suggests that the model's current feature extraction is sensitive to brightness and contrast changes, which is expected given the relatively small input size (48×48) and limited dataset diversity.

Color adjacency and feature ambiguity. The recurring confusion among specific color pairs (*grey↔blue*, *orange↔red*) indicates that chromatic proximity in hue space remains a challenge. These failures may be exacerbated by image resizing, which can desaturate subtle color differences and reduce discriminative feature quality. Moreover, texture or background cues that could help disambiguate similar colors are sometimes lost at low resolution.

Dataset size and diversity constraints. Although 5-fold cross-validation ensures robust evaluation, the dataset's limited size may not capture the full variability of Egyptian traffic conditions, including extreme weather, varied camera angles, and occlusions. As a result, certain edge cases are underrepresented, contributing to the observed misclassifications.

Opportunities for improvement. Several strategies could further enhance model performance and robustness:

- **Data augmentation:** Applying color jitter, histogram equalization, random brightness/-contrast adjustments, and synthetic illumination variations can help the model generalize better to real-world lighting conditions.
- **Preprocessing techniques:** Incorporating adaptive histogram normalization or color space transformations (e.g., HSV or Lab) could improve separation of adjacent hues.
- **Architecture enhancements:** Adding attention mechanisms or lightweight residual connections may allow the model to focus more selectively on informative regions without substantially increasing computational cost.
- **Dataset expansion:** Collecting more samples from diverse environments, including rare edge cases, would reduce model uncertainty and mitigate high-confidence misclassifications.

Summary. Overall, while the Simple CNN provides a good baseline with high accuracy and computational efficiency, targeted interventions in data augmentation, preprocessing, and minor architectural refinements have the potential to address the remaining errors and further strengthen real-world applicability. These improvements form a clear roadmap for future work, aiming to achieve near-perfect license plate color classification even under challenging traffic conditions.

5.2 Future Work

Building upon the identified limitations, future work may include:

- Exploring more attention-based or residual CNN architectures to further improve robustness against color ambiguity.
- Integrating adaptive preprocessing pipelines for real-time illumination correction.
- Collecting a larger, more diverse dataset capturing extreme lighting, occlusions, and varied camera angles.

- Deploying the model in real-world traffic monitoring systems and evaluating its performance under continuous streaming conditions.

6 Societal Impact

6.1 Potential Applications in Egyptian Traffic Management

The proposed license plate color classification system can support multiple traffic management applications in Egypt:

- **Automated traffic monitoring:** By classifying vehicle plate colors in real-time, authorities can streamline vehicle identification workflows, assisting in congestion analysis and traffic pattern studies.
- **Law enforcement support:** Rapid detection of vehicle types and special categories (e.g., commercial vs. private) through color cues can facilitate targeted enforcement of regulations.
- **Smart traffic infrastructure:** Integrating the model with camera networks can enable dynamic traffic signaling, adaptive tolling, or predictive maintenance of traffic systems based on vehicle flow patterns.

6.2 Privacy and Surveillance Considerations

While the system provides operational benefits, it also raises important ethical and privacy concerns:

- **Anonymization of personal data:** The model processes plate colors without necessarily storing sensitive personal identifiers. However, integration with full license plate recognition systems could inadvertently enable tracking of individual vehicles.
- **Data retention policies:** Strict policies are required to ensure that collected video or image data are stored only for necessary periods and used solely for legitimate traffic management purposes.
- **Transparency and public consent:** Deployment should consider public awareness campaigns and transparency reports to maintain trust and comply with local regulations regarding surveillance.

6.3 Bias and Fairness Analysis Across Vehicle Types

Although the model focuses on color classification rather than full identification, fairness considerations remain crucial:

- **Vehicle type imbalance:** Certain vehicle categories may be over- or underrepresented in the dataset, potentially affecting model performance.
- **Lighting and environmental conditions:** Biases may arise if the model performs better under conditions typical for certain vehicle types, leading to systematic disparities in accuracy.
- **Mitigation strategies:** Employing stratified data collection, synthetic augmentation for underrepresented types, and continuous monitoring of model errors across vehicle categories can reduce fairness gaps.

Overall, while the proposed model can improve traffic operations, careful attention to privacy, surveillance ethics, and fairness is essential to ensure responsible deployment in public environments.

7 Conclusion

This work presented the development and evaluation of a license plate color classification system tailored to Egyptian traffic conditions. The dataset was meticulously curated from diverse urban contexts, spanning administrative districts, tourist zones, and main roads, with temporal variations from morning to night. Such diversity ensured a representative distribution of vehicle types and environmental conditions, enhancing the dataset's applicability for real-world classification tasks.

Manual frame extraction was employed to maximize data quality, capturing even rare plate categories and enabling context-aware labeling. A rigorous train-test split strategy, combined with targeted data augmentation, addressed class imbalance while preserving the critical color information necessary for accurate classification.

Three deep learning architectures - Simple CNN, EfficientNet-B0, and MobileNetV2 - were systematically evaluated using 5-fold cross-validation. Quantitative analysis demonstrated that the custom-designed Simple CNN achieved the best trade-off between accuracy, F1-score, and computational efficiency, outperforming more complex pretrained networks on this low-resolution, color-sensitive dataset. Error analysis revealed that residual misclassifications were primarily due to illumination variability, color adjacency, and limited representation of extreme edge cases. These insights informed recommendations for targeted augmentation, preprocessing, and potential architectural enhancements.

Finally, the study emphasizes the practical utility of the proposed system for traffic management applications in Egypt, including automated monitoring and enforcement, while also highlighting ethical considerations regarding privacy and fairness. Overall, the work establishes a robust baseline for license plate color classification and provides a clear roadmap for future improvements, aiming toward near-perfect real-world performance under diverse traffic conditions.

References

[1] A. Ramadan, A. Ali, and F. Ramadan, "Real-time egyptian license plate detection and recognition using yolo," *International Journal of Advanced Computer Science and Applications*, vol. 13, 01 2022.

[2] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," 2020.

[3] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2019.