# Spill Detection System: Project Report

## Executive Summary

This project implements a computer vision system for detecting liquid spills on floors using YOLOv8 object detection. The system includes both a primary model (YOLOv8m) and a lightweight model (YOLOv8n) for different deployment scenarios. Through data augmentation and transfer learning techniques, we achieved high performance metrics, with the lightweight model outperforming the primary model in detection accuracy while being significantly faster and smaller.

## Introduction

Liquid spills in industrial and commercial environments pose significant safety risks, leading to slip and fall accidents. Automated detection systems can help identify spills quickly, enabling prompt cleanup and reducing accident risks. This project develops a computer vision solution that can detect spills in real-time using object detection.

## Dataset

The project uses a custom dataset consisting of 44 annotated images of liquid spills on various floor surfaces. The annotations are in YOLO format, providing bounding box coordinates for each spill instance.

### Dataset Augmentation

To address the limited size of the original dataset, we implemented comprehensive data augmentation techniques:

1. **Spatial Transformations**:
    - Horizontal and vertical flips
    - Random rotations (up to 90 degrees)
    - Shift, scale, and rotate transformations
2. **Visual Adjustments**:
    - Brightness and contrast variations
    - Gamma adjustments
    - Hue, saturation, and value modifications
3. **Noise and Quality Variations**:
    - Gaussian noise addition

    - Gaussian blur
    - ISO noise simulation

4. **Environmental Simulations**:
    - Rain effects
    - Shadow overlays
    - Fog simulation

These augmentation techniques expanded our training dataset from 41 to 146 images (a 256% increase), significantly improving model generalization and robustness to various environmental conditions.

# Methodology

## Model Architecture

- **Primary Model**: YOLOv8m - A medium-sized version of YOLOv8 that offers a good balance between performance and computational requirements.
- **Lightweight Model**: YOLOv8n - A nano version that's significantly smaller and faster than the medium model, making it suitable for edge devices.

## Data Processing

- Images are resized to 640×640 pixels for the primary model and 416×416 for the lightweight model.
- Annotations are in YOLO format: `<class> <x_center> <y_center> <width> <height>` with normalized coordinates.
- The augmented dataset was split into training (80%) and validation (20%) sets.

## Training Strategy

- The primary model was trained using transfer learning from a pre-trained YOLOv8m model.
- The lightweight model was also trained using transfer learning from a pre-trained YOLOv8n model.
- AdamW optimizer with learning rate of 0.001 and weight decay of 0.0005.
- Early stopping with patience of 20 epochs was implemented to prevent overfitting.
- Training was performed on the augmented dataset to improve model generalization.

# Results and Performance Metrics

After training on the augmented dataset, both models were evaluated on the validation set using standard object detection metrics.

## Quantitative Results

| Metric | Primary Model (YOLOv8m) | Lightweight Model (YOLOv8n) | % Difference |
|---|---|---|---|
| Precision | 0.976 | 0.980 | +0.4% |
| Recall | 0.750 | 1.000 | +33.3% |
| mAP50 | 0.856 | 0.973 | +13.7% |
| mAP50-95 | 0.689 | 0.857 | +24.4% |
| Inference Time (s) | 0.163 | 0.043 | -73.7% |
| Model Size (MB) | 49.58 | 5.92 | -88.1% |

## Key Observations

1. **Lightweight Model Performance**: The YOLOv8n model outperformed the YOLOv8m model in all detection metrics, particularly in recall (33.3% improvement) and mAP50-95 (24.4% improvement).

2. **Efficiency Gains**: The lightweight model demonstrated significant advantages in deployment metrics:

    - 3.8x faster inference time
    - 88.1% smaller model size (5.92MB vs 49.58MB)

3. **Impact of Data Augmentation**: The diverse augmentation techniques significantly contributed to model performance, especially for the lightweight model, allowing it to generalize better despite its lower capacity.

# Discussion

## Unexpected Results

The superior detection performance of the lightweight model was unexpected, as typically larger models achieve better accuracy. This can be attributed to several factors:

1. **Better Fit for Dataset Size**: The smaller model may be less prone to overfitting on the limited dataset, even with augmentation.

2. **Effective Data Augmentation**: The comprehensive augmentation strategy provided enough diversity in the training data to enable the lightweight model to generalize effectively.

3. **Simpler Problem Space**: The spill detection task may not require the capacity of larger models, as the visual features of spills are relatively straightforward.

## Practical Implications

The results have important practical implications for spill detection systems:

1. **Edge Deployment**: The lightweight model's significantly smaller size and faster inference make it ideal for deployment on edge devices with limited computational resources.

2. **Real-time Performance**: With 0.043s inference time (approximately 23 FPS), the lightweight model is suitable for real-time detection applications.

3. **Resource Efficiency**: The smaller model requires less memory and computational power, reducing hardware costs and energy consumption for deployed systems.

# Conclusion

This project demonstrates that effective data augmentation can significantly improve object detection performance, especially when working with limited training data. The superior performance of the lightweight model highlights the importance of matching model complexity to the specific application requirements.

The spill detection system achieves high accuracy (98% precision, 100% recall for the lightweight model) while maintaining efficiency suitable for real-world deployment. The data augmentation techniques proved critical in achieving these results with a limited original dataset.