# Improved Presentation Script with Q&A

Presentation Script: Shoplifting Detection Using YOLOv5 and ML

Slide 1: Title

Good [morning/afternoon], everyone. My name is Peerawit Wettayakorn, and today I’ll be presenting my project on shoplifting detection using YOLOv5 and machine learning.

Slide 2: Introduction to the Problem

Shoplifting causes billions in losses annually. Many small shops lack the resources for high-end surveillance solutions. My project aims to create a low-cost system that can automatically detect suspicious behavior from video footage.

Slide 3: Project Goals

The goals are: first, detect suspicious behavior using machine learning. Second, leverage YOLOv5 for real-time object detection. And third, ensure that this system is affordable and practical for small retail environments.

Slide 4: Tools & Technologies Used

The system uses Python, OpenCV for frame extraction, YOLOv5 for object detection, and machine learning models such as RandomForest, XGBoost, LightGBM, and a stacked model for behavior classification.

Slide 5: Dataset & Labeling Process

I used videos from the DCSASS dataset and a Kaggle shoplifting dataset. Frames were extracted using OpenCV, and auto-labeled using YOLOv5 based on whether the filename indicated shoplifting and if a person was present in the frame.

Slide 6: Classification Models

For the classification step, I used RandomForest, XGBoost, LightGBM, and a stacked ensemble model. Features included object counts, overlaps, and rolling averages for temporal smoothing.

Slide 7: XGBoost Here’s what we found.

XGBoost achieved 66% accuracy with a ROC-AUC score of 0.73. The model was decent, but still had room for improvement in distinguishing subtle behaviors.

Slide 8: LightGBM Here’s what we found.

LightGBM performed slightly better, reaching 73% accuracy and a ROC-AUC score of 0.83. It trained quickly and handled the tabular features very well.

Slide 9: RandomForest Here’s what we found.

RandomForest was similar to XGBoost, with 66% accuracy and ROC-AUC of 0.73. It was slightly less efficient than LightGBM.

Slide 10: Stacked Model Here’s what we found.

The stacked model combined all three base models and achieved 74% accuracy and a ROC-AUC score of 0.83. This shows that combining multiple models can improve robustness.

Slide 11: Limitations & Challenges

Some challenges included noisy labels from the auto-labeling process, a limited variety of shoplifting behavior in the data, and the lack of true motion-based features like tracking.

Slide 12: Real-World Application

This system can be deployed in small retail shops. It can monitor security footage and alert staff in real-time when suspicious behavior is detected. It’s also extendable to detect other behaviors.

Slide 13: Future Work

Future improvements include adding temporal tracking across frames, training on longer sequences, and integrating real-time alert systems. I’d also like to fine-tune on more diverse and labeled datasets.

Slide 14: Thank You / Q&A

Thank you for your time. I’m happy to take any questions you might have.

Q&A Predictions

Q: Why did you choose YOLOv5 over YOLOv8 or other detectors?

A: YOLOv5 is widely supported, easier to integrate, and has pre-trained models with good performance on common object classes.

Q: How did you handle class imbalance?

A: I used oversampling of the minority class using scikit-learn's resample function to ensure the models don’t overfit to the normal class.

Q: How real-time is your system?

A: Frame extraction and detection can be done in near real-time. However, classification runs after each frame or set of frames, so there’s slight latency. Optimizations are possible.

Q: What are your plans to improve accuracy?

A: I plan to add temporal tracking features, use optical flow or other motion analysis techniques, and possibly add LSTM models for time-series behavior modeling.

Q: How generalizable is this system?

A: The current model is trained on limited datasets, so generalization is a concern. It needs more diverse data and real-world testing to improve reliability.

Q: What’s the biggest technical challenge you faced?

A: Dealing with noisy and weak labels from auto-labeling was the most difficult. Also, extracting meaningful behavior from a single frame without context is inherently limited.