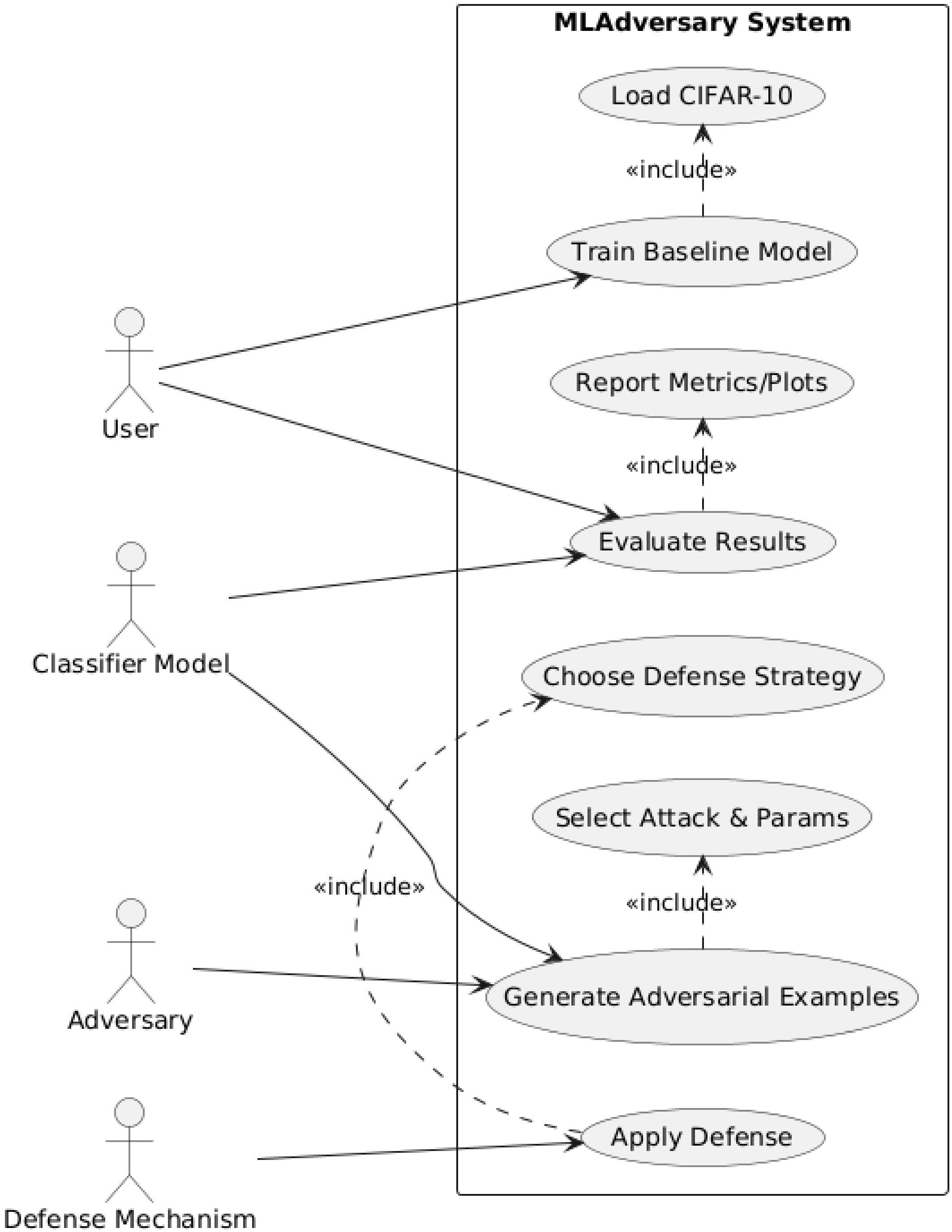
CptS 528 Advanced Cyber Security

Fall 2025

Team: Ye_MLAdversary

Project Deliverable 2-1

Use Case Diagram



Use Case Name

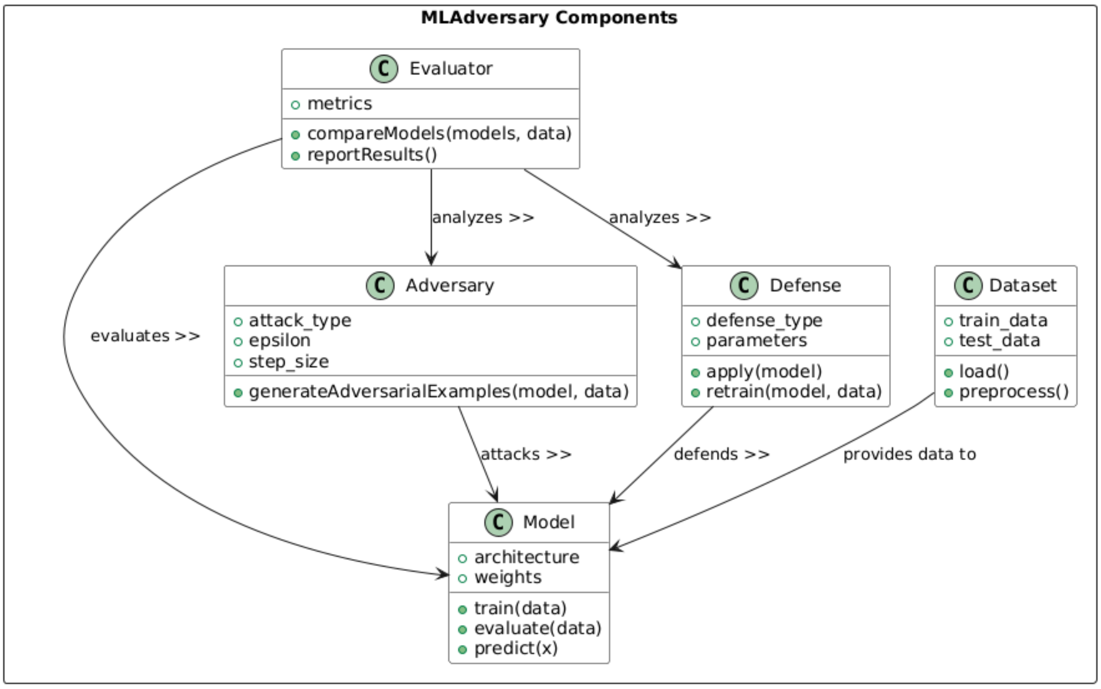
Use Case Name	Train Baseline Model
Actors	User
Preconditions	CIFAR-10 dataset available and ready to be trained
Goal	Train a baseline classifier for the dataset
Scenario	1. Load CIFAR-10 dataset 2. Initialize model 3. Train the model for N epochs 4. Save the weights
Exception	- Memory error - Dataset mismatch

Use Case Name	Generate Adversarial Example
Actors	Adversary, Classifier Model
Preconditions	Baseline model available and test set are ready
Goal	Create adversarial inputs that cause misclassification
Scenario	1. Select attack and parameters 2. Compute gradients 3. Run inference on adversarial set 4. Record attack success rate
Exception	- Gradient disabled - Incompatible attack settings

Use Case Name	Apply Defense
Actors	Defense Mechanism
Preconditions	Adversarial examples available
Goal	Improve efficiency to chosen attacks
Scenario	1. Choose defense 2. If training: mix clean batches with adversarial batches and retrain 3. If preprocessing: apply transform at inference 4. Evaluate the accuracy under same attack settings
Exception	- Defense overly degrades clean accuracy - Training time takes too long

Use Case Name	Evaluate Results
Actors	User, Classifier model
Preconditions	Baseline, attacked, and defended models/outputs available
Goal	Compare clean/adversarial/defended performrmance
Scenario	1. Run standardized evaluation on clean test set 2. Aggregate metrics 3. Produce plots/tables and store seeds/configs
Exception	- Non-reproducible runs due to missing seeds - Corrupted logs

UML Diagram



Quality Plan

Security Goals

- Confidentiality
    - Prevent unauthorized access to training data and adversarial examples
    - Ensure that only legitimate users can view experiment logs and results
        - Restrict dataset and model access to authenticated users
        - Encrypt logs and outputs
- Integrity
    - Ensure that adversarial perturbations and defenses are correctly implemented without corrupt the data
    - Guarantee that evaluation metrics reflect actual model performance

without tampering
        - Use checksums for dataset integrity, seed management for reproducibility
        - Enforce immutability in model weights after training checkpoints
- Availability
    - Ensure the trained model remains usable even under adversarial conditions
    - guarantee that evaluation and defense mechanisms can run reliably without causing denial-of-service due to excessive computation
        - optimize attacks and defenses to run within bounded resources
        - support fallback to baseline evaluation if defenses fail

Security Metrics

- Confidentiality
    - Percentage of experiment files or model weights access by unauthorized users
        - Target: 0%. > 0% indicates a confidentiality breach
- Integrity
    - Percentage of evaluation runs where outputs deviate due to corrupted datasets or logs
        - Targets: 0%. >0% indicates dataset corruption or log tampering that compromises evaluation integrity
    - Percentage of adversarial samples that exceed defined perturbation limits
        - Targets: 0%. >0% indicates the adversarial generation process is violating security constraints and producing invalid examples
- Availability
    - Percentage of evaluation jobs that fail to complete due to resource exhaustion
        - Target: <15% failed jobs. >15% indicates the system is too resource-intensive under adversarial conditions and fails to maintain availability
    - Average runtime overhead introduced by defenses relative to baseline inference
        - Target: <2 times the baseline runtime. >2x indicates defenses degrade system availability by imposing excessive computational cost