## **CS 781: Formal Methods in Machine Learning**

**project plan**

**Topic 2:  
Iterative, Robustness-Guided Data Augmentation for a CNN-based Car Detector using VerifAI and α,β-CROWN**

This project proposes the design, implementation, and evaluation of a closed-loop system for improving the verifiable robustness of a Convolutional Neural Network (CNN) for object detection. The central hypothesis is that an iterative re-training process, guided by formal robustness verification metrics, can produce a model that is demonstrably more robust against adversarial perturbations than one trained using conventional data augmentation techniques.

The system will operate in a feedback loop. The [VerifAI](https://verifai.readthedocs.io/en/latest/) toolkit, utilizing the [Scenic](https://scenic-lang.org/cvpr24/) probabilistic programming language, will generate synthetic training data consisting of images of cars on roads. A CNN model will be re-trained on this data. Subsequently the [α,β-CROWN](https://github.com/Verified-Intelligence/alpha-beta-CROWN) formal verification tool will be employed to compute a precise, quantitative measure of the re-trained network's local robustness on specific test cases. The core innovation of this project lies in using these quantitative robustness metrics to programmatically inform and adapt the scenic data sampling distribution for the subsequent iteration of data generation and re-training. This approach refines the paradigm of counterexample-guided data augmentation, where the concept of a "counterexample" is elevated from a mere misclassification to a quantifiable deficiency in local robustness. The project aims to demonstrate a monotonic improvement in the network's certified robustness over multiple iterations, providing empirical evidence for the efficacy of this verification-driven training methodology.

**Team:**

| Name | Roll No | Email Id |
| --- | --- | --- |
| Arin Weling | 22B1230 | 22b1230@iitb.ac.in |
| Aman Moon | 22B1216 | 22b1216@iitb.ac.in |

**Deliverables:**

The core deliverable for this project is a demonstration of the iterative re-training and robustness verification process. Specifically, we will:

* Use the VerifAI toolkit to re-train a simple convolutional neural network for detecting images of cars on roads. This re-training will occur in distinct iterations.
* At the end of each iteration, after the network has been re-trained, we will use α,β-CROWN to determine the local robustness of the re-trained network for a given image. This will be measured as the radius of an L\_p norm ball (for a suitable "p" of your choice) for which the class label of the image remains unchanged.
* For every iteration, we will present the model example and its output. To demonstrate the verification process, we will provide the corresponding detailed results from α,β-CROWN. This will include the key parameters of the analysis, such as the specific L\_p norm and perturbation radius (ϵ) used, along with the verification outcome, which includes the final status ('verified', 'falsified', or 'timeout') and the computed lower bound that certifies the model's local robustness.
* We will present the output from every model after every iteration.

**Anticipated Technical Challenges:**

* Steep Learning Curve: we will need to learn three highly specialized, academic toolkits: VerifAI, Scenic, and α-β-CROWN. Unlike mainstream libraries, they have smaller user communities and less extensive documentation, which means solving problems will require more independent research and experimentation.
* We might face some challenges in increasing the robustness by guiding the Scenic sampler to generate new training data based on the **α-β-CROWN** analysis.
* The change in robustness value might not be prominent.

**Primary Implementation Strategy:**

* **Learn the Tools:** Start by using YouTube resources to understand the concepts of **α-β-CROWN**.
* **Create a Baseline:** Implement an initial CNN model and use Alpha-Beta-CROWN to measure its baseline robustness on a single test image.
* **Begin Iterative Loop:** Start the retraining process, which will happen in iterations.
* **Retrain Intelligently:** In each iteration, use the robustness results to guide the **Scenic** sampler, generate new training data with **VerifAI**, and then retrain the model.
* **Verify and Track:** After each retraining cycle, re-verify the model's robustness to track improvement. The goal is to show the robustness score increases with each iteration.

**Contingency Strategy:**

## **Scenic Sampler Challenges:** If guiding the Scenic sampler to generate meaningful training data based on robustness feedback becomes infeasible, we will adopt a simplified approach: manually biasing data generation parameters (e.g., lighting, positioning, perturbations) to simulate adversarial conditions.

## **Robustness Gains Not Significant:** If iterative re-training does not show noticeable improvement in robustness, we will (i) benchmark against standard adversarial training baselines, (ii) adjust iteration parameters (e.g., learning rate, sample size, number of iterations), and (iii) present comparative results to still demonstrate the effect of verification-driven training.

## **Timeouts or Failures in α-β-CROWN:** If the verification step frequently times out or fails, we will (i) restrict evaluation to smaller test cases, (ii) switch to approximate robustness estimators for interim validation, and (iii) report both verified and approximated results transparently.

## **Timeline Slippage:** If any milestone cannot be met on schedule, the buffer period (1–5 Nov 2025) will be used. If delays extend beyond the buffer, non-critical extensions (such as additional iterations) will be scaled back while ensuring core deliverables (baseline model, at least one full iteration, and verification results) remain intact.

## **Project Timeline and Milestones:**

| Sr. No. | Timeline | Primary Goal | Deliverable |
| --- | --- | --- | --- |
| 1 | 1rd Oct 2025 -  5th Oct 2025 | Environment Setup & Tool Tutorials | Install all required toolkits and get familiar with the libraries to be used. |
| 2 | 5th Oct 2025 -  8th Oct 2025 | Baseline Component Implementation | Implement the baseline CNN and a static Scenic script. |
| 3 | 9th Oct 2025 -  13th Oct 2025 | Initial Training | Train the CNN and create a base detection model. |
| 4 | Mid Oct | Check-Point |  |
| 5 | 14th Oct 2025 -  30th Oct 2025 | Begin the Iterative loop | Use alpha-beta-crown's analysis to guide **Scenic** in generating new data, then retrain the model to iteratively increase its verifiable robustness. |
| 6 | Early Nov | Check-Point |  |
| 7 | 1st Nov 2025 -  5th Nov 2025 | Buffer Period |  |
| 8 | 6th Nov 2025 -  10 Nov 2025 | Results Analysis & Final Submission | Analyze full experimental results, prepare the final report, and create the demo.  Final project submission (code, report, presentation). |
| 9 | 24 Nov 2025 | Final Presentation |  |

## **Updates:**

**We have already initiated Part 1 of the project, completed the environment setup, and are maintaining the updated codebase at the link provided below:**

[**https://github.com/arinweling/cs781/tree/main**](https://github.com/arinweling/cs781/tree/main)