AdvSecureNet Master's Project Description

1 Introduction

Machine learning models have become integral to numerous domains, including self-driving cars [1], facial recognition [11, 6], medical imaging [8], and natural language processing tasks like chatbots [2] and translation services [12]. Despite their widespread adoption, these models are susceptible to adversarial attacks—subtle input manipulations that can deceive the models [5, 13]. Such vulnerabilities can compromise the integrity, confidentiality, or availability of systems relying on these models [7].

Existing libraries like ART [9], AdverTorch [4], and CleverHans [10] offer tools for exploring these vulnerabilities through attacks, defenses, and evaluation metrics. However, they often lack critical features necessary for comprehensive research and experimentation, such as native multi-GPU support, integrated command-line interfaces (CLI) and application programming interfaces (API), and support for external configuration files.

To bridge these gaps, we developed AdvSecureNet [3] (Adversarial Secure Networks) , a comprehensive and flexible Python toolkit optimized for multi-GPU setups. It supports various adversarial attacks, defenses, and evaluation metrics and includes both CLI and API interfaces, offering versatile options for experimentation and research. AdvSecureNet is available as an open-source project on GitHub at `https://github.com/melihcatal/advsecurenet`. However, as of now, the toolkit is limited to computer vision classification tasks. This Master's project aims to expand AdvSecureNet's capabilities to include:

- Natural Language Processing (NLP) Attacks and Defenses
- Audio Recognition Attacks and Defenses
- Large Language Models (LLMs) Vulnerabilities
- Fairness and Bias Evaluation Metrics

By incorporating these extensions, we aim to create a comprehensive toolkit for trustworthy AI research across various domains. This project is a collaborative effort between the AIML and S.E.A.L. research groups. Prof. Dr. Manuel Günther from the AIML research group serves as the responsible professor, while Melih Catal from the S.E.A.L. research group acts as the main supervisor, managing the project's progress and execution.

2 Project Requirements and Responsibilities

2.1 Technical Skills

- **Software Engineering**: Experience in software development, unit testing, documentation, version control and CI/CD pipeline.
- **Machine Learning**: Understanding of machine learning concepts and experience with Python and PyTorch.
- **Interest in Responsible and Trustworthy AI**: Familiarity with adversarial attacks, defenses, and Trustworthy AI, or a willingness to learn.

2.2 Responsiblities

- **Implementation**: Implement features with high code quality and maintainability, ensuring compliance with the existing toolkit's architecture. This includes support for both CLI and API interfaces, external YAML configuration files, and multi-GPU setups. Apply key software engineering principles such as thorough documentation, rigorous testing, and the integration of a CI/CD pipeline to enhance development and deployment processes.
- **Research**: Read and understand relevant research papers to enhance understanding of the field and stay updated on current trends and advancements.
- **Collaboration**: Participate in the regular meetings with the main supervisor to discuss progress and challenges, and collaborate with team members to manage workload effectively and ensure timely completion of project tasks.

3 Proposed Extensions

Depending on the team size and interests, the project scope will be finalized. Below are potential extensions summarized in Table 1.

Task	NLP Extension	Audio Extension	LLM Extension	Fairness & Bias Extension
Implement Attacks	TextFooler, HotFlip	Audio Adversarial Examples, Hidden Voice Commands	Jailbreaking attacks, prompt injection attacks	—
Implement Defenses	Adapt existing adversarial training methods, develop new ones	Adversarial training, input transformation defenses, anomaly detection	Research and develop appropriate defenses	—
Impelement Evaluation Metrics	BLEU, ASR	To be determined	Toxicity, Bias	Statistical Parity Difference, Equal Opportunity Difference
Support Datasets	HuggingFace and custom datasets	Common Voice, LibriSpeech, VoxCeleb, HuggingFace support	HuggingFace support	Adult, COMPAS, CrowS-Pairs
Support Models	HuggingFace and custom models	DeepSpeech, Wav2Vec	HuggingFace models, API support	—
Implement Benign Training	✓	✓	✓	—

Table 1: Summary of Proposed Extensions

4 Schedule

Given that the project features will be selected based on team size and interests, the following is a general schedule outlining the key phases over the course of 15 weeks (assuming 30 hours of work per week).

- **Weeks 1–3: Project Setup and Planning**
  - Familiarization with the AdvSecureNet toolkit and existing codebase.
  - Literature review on adversarial attacks, defenses, and evaluation metrics in selected domains (e.g., NLP, Audio, LLMs, Fairness & Bias).
  - Selection of specific features to implement based on group size and interests.

  - Setting up the development environment, including version control and CI/CD pipelines.
  - Work environment creation, including access to GPU server.
  - Designing software diagrams to illustrate the architecture.
  - **Deliverable:** One-page project proposal outlining workloads, group members, expected start and finish dates, and selected features to be implemented.

- **Weeks 4–10: Implementation of Selected Extensions**
  - Implementing adversarial attacks in the chosen domain(s).
  - Developing corresponding defenses for the implemented attacks.
  - Ensuring compatibility with CLI and API interfaces, multi-GPU support, and supporting external YAML configuration files.
  - Supporting relevant datasets and models (e.g., HuggingFace models for NLP, DeepSpeech for audio).
  - Implementing evaluation metrics specific to the domain(s) (e.g., BLEU score for NLP, statistical parity difference for fairness).
  - Implementing benign (standard) training procedures for baseline comparisons, if needed.

- **Weeks 11–12: Testing and Documentation**
  - Rigorous unit testing to achieve at least 95% line coverage.
  - Comprehensive documentation of code and features.
  - **Note: It is better practice to incorporate these activities from the beginning of the project and to do them in parallel with the implementation.**

- **Weeks 13–15: Finalization, Reporting, and Presentation**
  - Final integration and testing of all implemented features.
  - Writing the final report, detailing implemented features, challenges, and future work.
  - Preparing the final presentation summarizing project outcomes.
  - Finalizing and submitting all project deliverables.
  - Presenting the project to supervisors and research groups.
  - **Deliverables:** Final report, presentation slides, and all implemented code and documentation.

5 Grading Criteria

- **Implementation (30%)**: Quality of code, unit testing (95% line coverage), documentation, and adherence to best practices, ensuring compliance with the existing toolkit's architecture and standards.
- **Final Report (40%)**: Detailed report on implemented features, challenges, lessons learned, and future work.
- **Presentation (30%)**: Final presentation summarizing the project outcomes.

6 Contact

- **Melih Catal** - catal@ifi.uzh.ch
- **Prof. Dr. Manuel Günther** - guenther@ifi.uzh.ch

References

[1] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[2] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *ArXiv*, abs/2005.14165, 2020.

[3] Melih Catal and Manuel Günther. Advsecurenet: A python toolkit for adversarial machine learning. *arXiv preprint arXiv:2409.02629*, 2024.

[4] Gavin Weiguang Ding, Luyu Wang, and Xiaomeng Jin. AdverTorch v0.1: An adversarial robustness toolbox based on pytorch. *arXiv preprint arXiv:1902.07623*, 2019.

[5] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[6] Manuel Günther, Laurent El Shafey, and Sébastien Marcel. Face recognition in challenging environments: An experimental and reproducible research survey. In *Face recognition across the imaging spectrum*. Springer, 2016.

[7] Faiq Khalid, Muhammad Abdullah Hanif, and Muhammad Shafique. Exploiting vulnerabilities in deep neural networks: Adversarial and fault-injection attacks. In *Proceedings of the Fifth International Conferenceon Cyber-Technologies and Cyber-Systems*, pages 24–29, 2020.

[8] Yoav Mintz and Ronit Brodie. Introduction to artificial intelligence in medicine. *Minimally Invasive Therapy & Allied Technologies*, 28(2):73–81, 2019.

[9] Maria-Irina Nicolae, Mathieu Sinn, Minh Ngoc Tran, Beat Buesser, Ambrish Rawat, Martin Wistuba, Valentina Zantedeschi, Nathalie Baracaldo, Bryant Chen, Heiko Ludwig, Ian Molloy, and Ben Edwards. Adversarial robustness toolbox v1.2.0. *CoRR*, 1807.01069, 2018.

[10] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, Rujun Long, and Patrick McDaniel. Technical report on the cleverhans v2.1.0 adversarial examples library, 2018.

[11] Divyarajsinh N Parmar and Brijesh B Mehta. Face recognition methods & applications. *International Journal of Computer Technology and Applications*, 4(1):84, 2013.

[12] Martin Popel, Marketa Tomkova, Jakub Tomek, Łukasz Kaiser, Jakob Uszkoreit, Ondřej Bojar, and Zdeněk Žabokrtskỳ. Transforming machine translation: a deep learning system reaches news translation quality comparable to human professionals. *Nature communications*, 11(1):4381, 2020.

[13] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, D. Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.