UC San Diego

Trojan AI Detection

Christopher Armstrong (cparmstr@ucsd.edu), Daniel Hartley (dhartley@ucsd.edu), Spencer Hutton (sphutton@ucsd.edu), Shirley Quach (shquach@ucsd.edu)
Advisors: XinQiao Zhang, Dr. Tara Javidi, Dr.Farinaz Koushanfar

June 9, 2023

Problem Statement

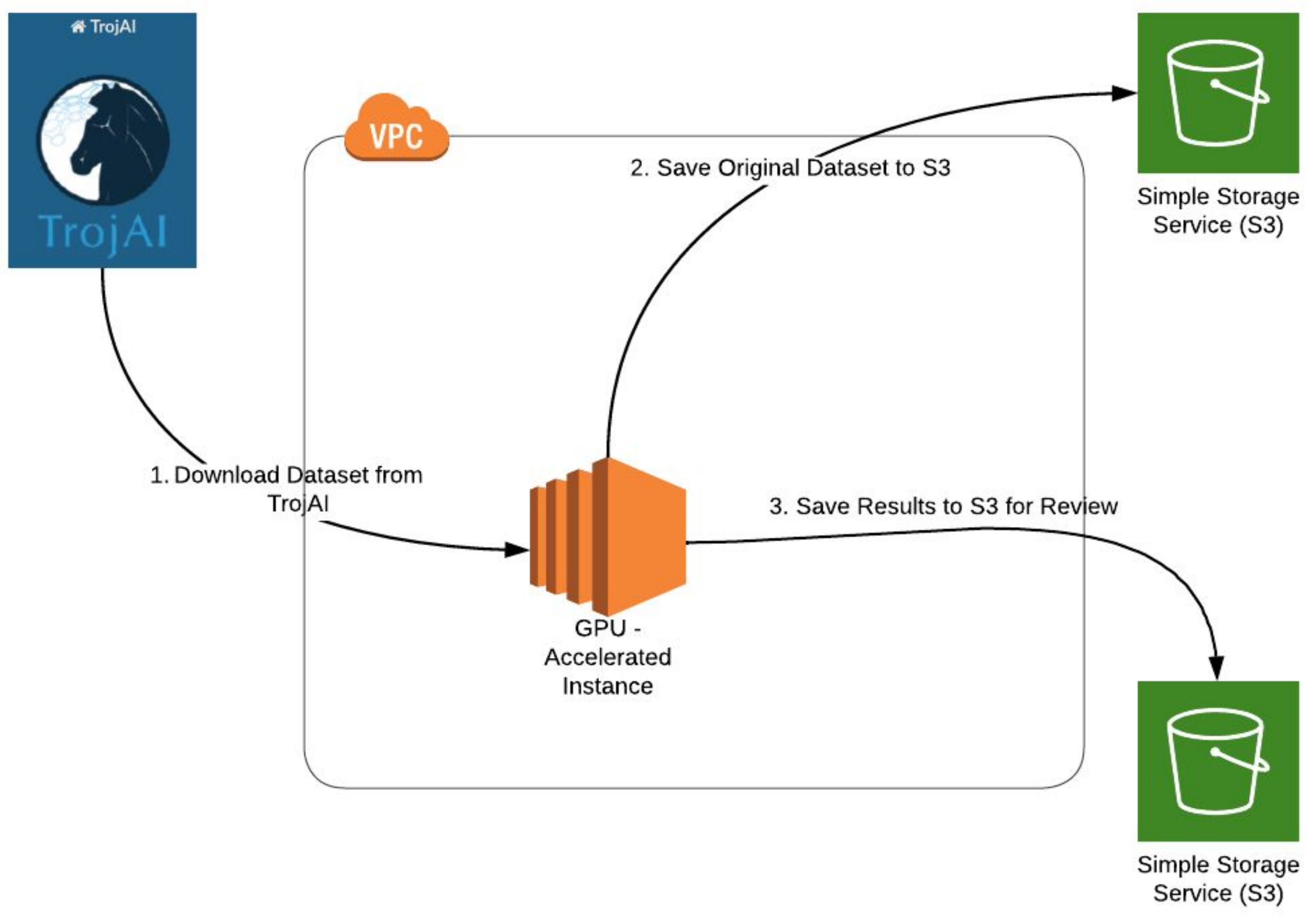
As machine learning (ML) gains prominence in the business world, the implementation of deep neural networks (DNN) has become more widespread. The security of DNN models has recently come under scrutiny as they are at risk of adversarial attacks such as backdoor Trojan attacks. These attacks depend on a trigger to activate malicious behavior. Due to the lack of transparency in DNNs, the effects of Trojans may remain undetected until activated by an attacker. This project demonstrates a significant reduction in the time and resources necessary to detect a poisoned model through the use of dimensionality reduction techniques. The detector utilizes Principal Component Analysis (PCA) and Independent Component Analysis (ICA) to reduce model weights that can then be used to train a classification model. This work builds on previous research, integrating reduction techniques to significantly reduce inference time while maintaining model accuracy at 85%. Are you protected from malicious AI?

Scalability and Robustness

- **Architecture agnostic**
  - Generate training data for your target models, feed to trainer
- **Built-in optimization**
  - Model hyperparameter tuning and reduction optimization is embedded into the trainer
- **Customization**
  - Detector training is configurable to include optimized detector architecture search
- **Integration**
  - 2 sec/model inference means flexible integration into cyber security strategy
- **Modular**
  - A modular solution allows businesses to integrate Trojan Detection into the AI development lifecycle

Detector Development

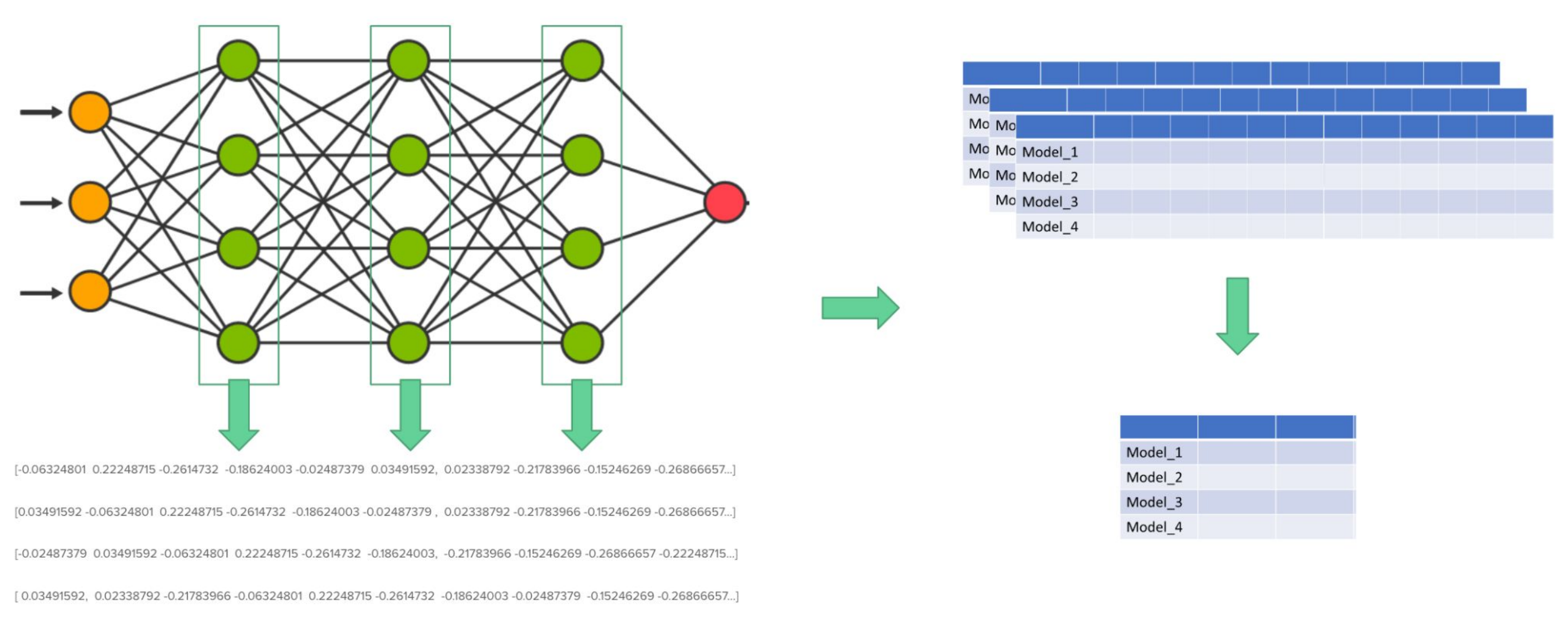
- **Leverage** cloud storage and GPU accelerated compute instances to support CUDA PyTorch requirements
- **Push** data to the cloud so the large dataset continues to be available for further testing
- **Save** the original data as well as the results to allow for follow-up analysis and reproducibility of the detector
- **Deploy** the trained detector to protect systems from Trojan AI


TrojAI
VPC
1. Download Dataset from TrojAI
2. Save Original Dataset to S3
3. Save Results to S3 for Review
GPU - Accelerated Instance
Simple Storage Service (S3)
Simple Storage Service (S3)

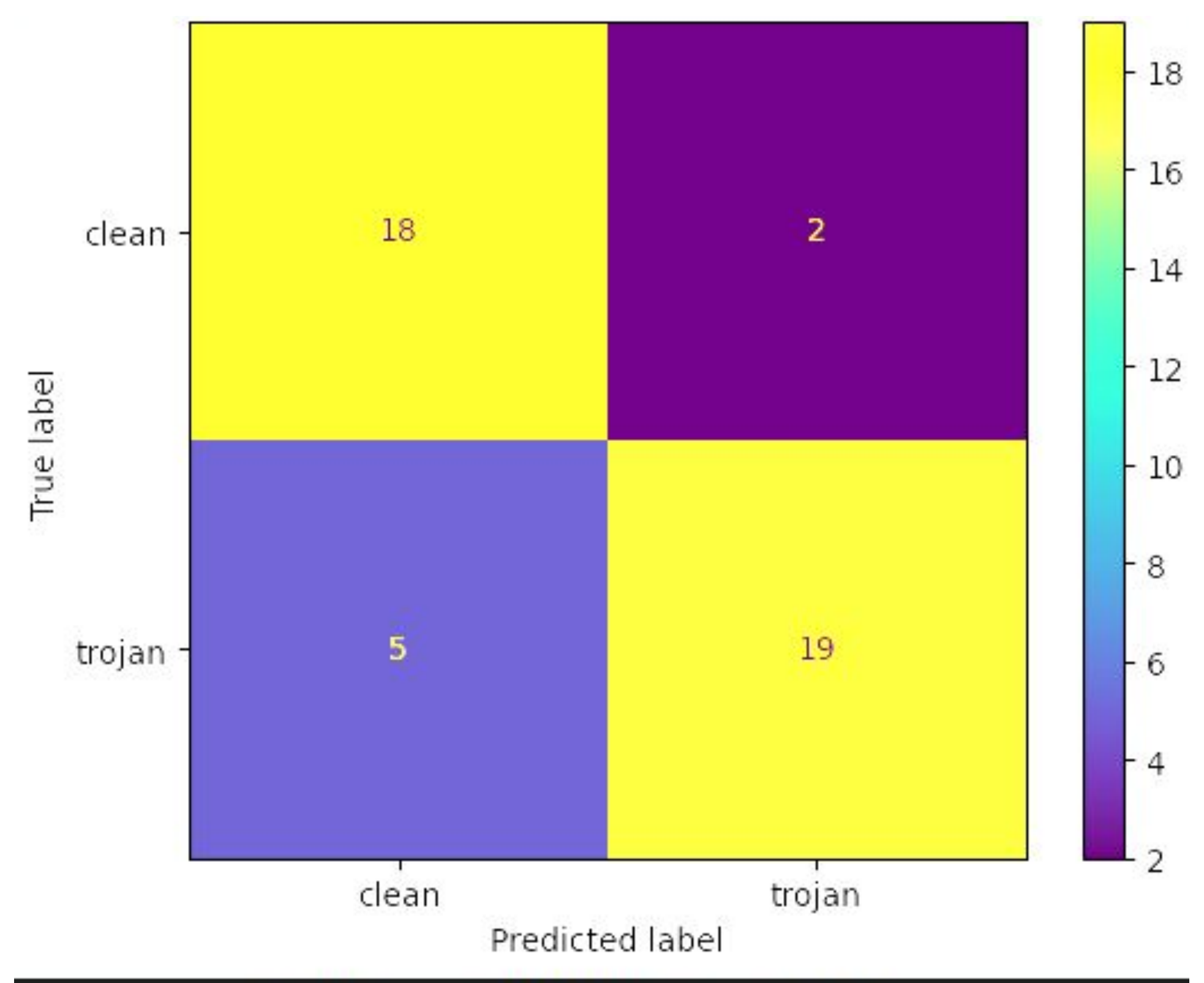
How Will We Detect Trojans?

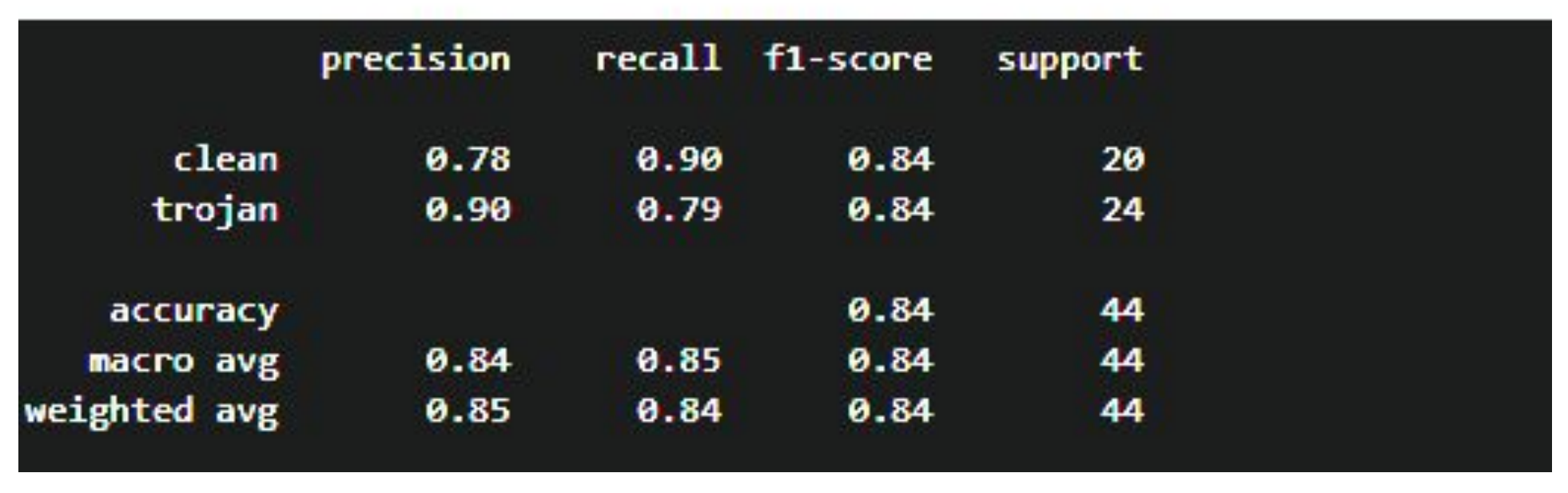
Trojan models may be detected by training a model to examine the model's weights. This method is fast, low cost and can be integrated into a more complex model detection strategy

1. **Each model layer is flattened and reduced**



2. **Feature Reduction**
   Multilevel reduction to reduce models to the same shape. Kernel PCA is applied for each layer and architecture. ICA is then applied on the reduced dataset
3. **Detector Training**
   An optimized search procedure selects the reduction parameters and hyper parameters and then the reduced model weights are used to train a classification model using XGBoost. Hyperparameters are tuned using Optuna
4. **Results**
   Model performance reaches 84% accuracy and inference duration is 2 sec/model. Top detectors on the NIST TrojAI leaderboard running the same dataset average 400 sec/model





```
              precision    recall  f1-score   support

       clean       0.78      0.90      0.84        20
      trojan       0.90      0.79      0.84        24

    accuracy                           0.84        44
   macro avg       0.84      0.85      0.84        44
weighted avg       0.85      0.84      0.84        44
```

Key Insights and Conclusion

Our model's performance has improved significantly compared to the earlier stages of our work and compared to the top results on the TrojAi leaderboards with performance accuracy reaching 84%. What originally took roughly 36 hours to complete, we were able to reduce the runtime to three hours for 150GB of modeled data.

Technology Stack

Python 3.10
Numpy
Tensorflow 2.10
CUDA 12
CUDNN 8
AWS

References

Backdoor Attack Detection in Computer Vision by Applying Matrix Factorization on the Weights of Deep Networks (2022)
https://arxiv.org/pdf/2212.08121v1.pdf
ABS: Scanning Neural Networks for Back-doors by Artificial Brain Stimulation
https://people.cs.rutgers.edu/~sm2283/papers/CCS19.pdf
Neural Cleanse: Identifying and Mitigating Backdoor Attacks in Neural Networks
https://sites.cs.ucsb.edu/~bolunwang/assets/docs/backdoor-sp19.pdf
An Embarrassingly Simple Approach for Trojan Attack in Deep Neural Networks
https://dl.acm.org/doi/pdf/10.1145/3394486.3403064
Michael Paul Majurski (2020), Trojan Detection Software Challenge - image-classification-jun2020-train, National Institute of Standards and Technology, https://doi.org/10.18434/M32195
https://data.nist.gov/od/id/mds2-2195
BadNets: Identifying Vulnerabilities in the Machine Learning Model Supply Chain
https://arxiv.org/pdf/1708.06733.pdf