Classification of DDoS Attacks using Deep Learning Techniques based on CICEV2023 Dataset

CS658 Course Presentation
Group 8

SAUGAT KANNOJIA

SANKALP PANDE

ROHIT KUMAR

MANASVI JAIN

Group Composition

Team Contributors:	
Saugat Kannojia	Data preparation, Cleaning and Preprocessing
Sankalp Pande	Data Conceptualization
Rohit Kumar	Data Analysis
Manasvi Jain	Model Training
Collaborative involvement	Report Creation and Presentation, Research on Data Optimization

Problem Statement:

To develop a DDoS attack detection model, trained on the **CICEV2023** dataset using system information, that accurately classifies whether the current system with the features created underwent a DDoS attack in EV charging networks, ultimately enhancing the resilience and operational stability of these critical systems.

Dataset:

The CICEV2023 dataset includes rich and diverse information capturing the behavior and activities of multiple EV charging stations, covering a wide range of parameters that reflect both attack and non-attack scenarios within the charging infrastructure.
These are 4 types of attack scenarios possible:-

- **Correct_ID** : The attacker tries to authenticate himself by obtaining a normal ID but does not have the correct key
- **Wrong_CS_TS** : The attacker changes the timestamp value between CS and GS to an old value, causing authentication failure at GS
- **Wrong_EV_TS :** The attacker causes authentication failure in CS by changing the timestamp value between EV and CS to an old value
- **Wrong_ID :** The attacker attempts authentication without the correct ID and the legitimate key.

Sub-division of Dataset

For each of the above, the dataset tries to create certain variations in the data :

- **Random_CS_Off** : The full attack mode attacks many identical EVs to all CSs simultaneously
- **Random_CS_On** : The random attack mode arbitrarily chooses the victim CS under the attacks
- **Gaussian_On** : Data with the Gaussian analysis to create a distribution similar to the normal EV authentication distribution
- **Gaussian_Off** : Data without the Gaussian attack strategy belongs in this directory

Dataset Description

- Data was in two forms: Raw and Processed
- Raw Data had txt files of Linux Kernel Overheads, System Performance Data proving to be hard to process
- Processed Data was used by us since it had organised data in form of directories
- Directory structure was heavily nested so we decided to make a few CSV files

- Each variety of attack scenario had 3 JSON files: TOP, STAT and TIME_DELTA
- TIME_DELTA: Interval between the previous and subsequent authentications
- STAT: No. of systemwide consumed cycles, instructions and branch instructions in CS
- TOP: Linux Kernel Overhead of various metrics observed in different scenarios

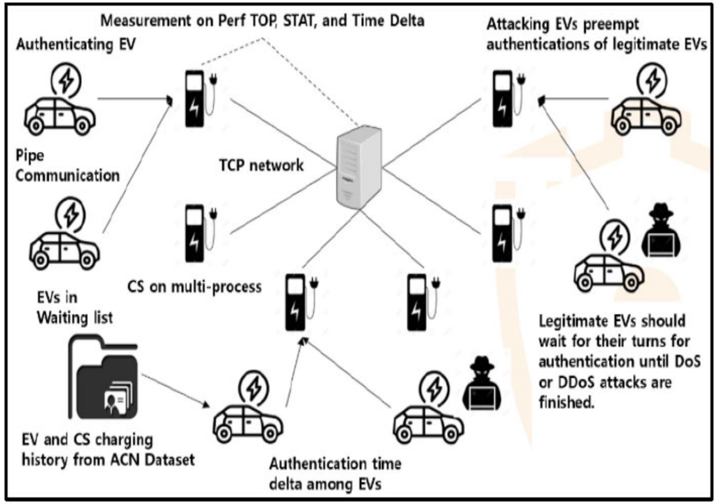
Views on the Data

- The environment set up for this is very specific for the data; it was not very diverse
- Dataset was very complex to deal with; the creation of features from the data proved to be a challenging task
- Data has some imbalance in Linux Kernel Overheads since it had a lot more metrics in attack scenario than normal

OS: Ubuntu 22.04.1 LTS (kernel version: 5.15.58)
Language: Python 3.10.6
CPU: Intel(R) Core(TM) i5-8265U @ 1.60GHz
RAM: 16GB
Perf: 5.15.74

Importance of this Problem for Cybersecurity

- Growth in EV Charging Infrastructure: With the rapid expansion of electric vehicle (EV) charging stations, cybersecurity concerns around these systems are rising
- DDoS Threat to EV Authentication Systems : Distributed Denial of Service (DDoS) attack targeting this system can disrupt authentication, potentially preventing EVs from accessing charging services



Feature Engineering

- Data had arrays of data points for different timestamps
- Statistical features like max, min, mean, standard deviation created
- Generated CSV Files for all scenarios and combined them into consolidated CSV files
- Performed Left Join on TOP data w.r.t to both STAT and TIME_DELTA to combine

Data Preprocessing

- NULL Values present in TOP data removed
- Applied Standard Scaling on the final features to make the model converge faster
- Balanced the no. of metrics in attack and normal scenarios for TOP Data
- Removed unnecessary features like sampling count, sampling resolution

Solution Methodology

Model Selection:

- A Feed-Forward Network (FFN) was chosen for its flexibility and ability to generalize well across various data splits.
- The network architecture includes:
  - Input layer
  - Two hidden layers with ReLU activation
  - Sigmoid output layer for binary classification

Cross-Validation:

- 5-fold cross-validation with 5 random splits was employed for consistent model evaluation across different data partitions.
- Key performance metrics (e.g., F1 score and accuracy) were averaged to establish a reliable benchmark for model performance.

Model Evaluation

Prevention of Overfitting:

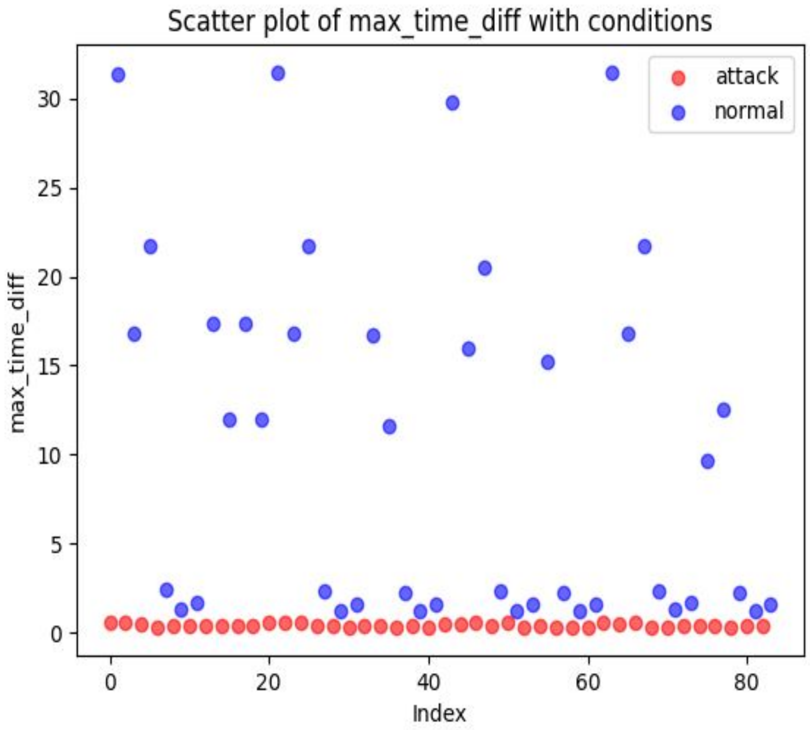
- Dropout layers were introduced in the architecture to prevent overfitting and improve generalization on unseen data
- Training showed a consistent decrease in both training and validation loss, indicating no overfitting

Optimization:

- Both Adam and SGD optimizers were considered
- Adam was selected as it showed superior performance

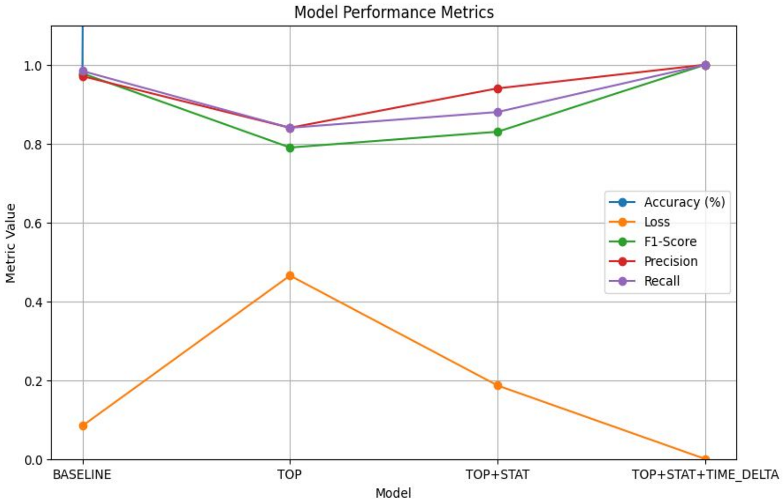
Novelty

- Creation of statistical features for the data to evaluate performance
- Performed analysis on all available data as compared to the baseline which just used R-OFF, G-OFF for training
- Combined TOP, STAT and TIME_DELTA features to get better classification



Results

Model	Data	Accuracy	Loss	F1-score	Precision	Recall
0	BASELINE	0.976	0.084	0.978	0.971	0.984
1	TOP	80.04±2.01	0.4652	0.79	0.84	0.84
2	TOP+STAT	92.56±1.0	0.1863	0.83	0.94	0.88
3	TOP+STAT+ TIME_DELTA	100±0.0	0.0000	1.00	1.00	1.00



Limitations

- The data provided to us was limited and didn't observe a large number of Charging Stations which were limited to just 3 in our case.
- The processed data was generated using some unknown techniques that were not provided to us, thus limiting us with the type of data extracted.
- This method cannot be used for real-time setting since the data is analysed after extracting it from the system.
- Any new types of Zero Day attack scenarios cannot be handled by our proposed model since it is heavily trained on this data.

Future Improvements

- Could explore methods of trying to do a realtime analysis for this type of situation
- Generate more data for different systems and perform Perf analysis on them to gain generality
- Create better initial features as compared to the statistical features extracted

References

- DDoS detection in electric vehicle charging stations: A deep learning perspective via CICEV2023 dataset
- https://www.unb.ca/cic/datasets/cicev2023.html

Libraries

- scikit-learn
- tensorflow
- json
- numpy
- pandas
- matplotlib

Learning Outcomes

- Learned how to clean, preprocess, and handle complex datasets, focusing on extracting relevant features for DDoS detection
- Learned how system information can be used to detect whether an attack is present or not

Thank You