THE UNIVERSITY OF TEXAS AT DALLAS

GCI: A GPU Based Transfer Learning Approach for Detecting Cheats of Computer Game

Md Shihabul Islam, Bo Dong, Swarup Chandra,

Faculty: Latifur Khan, PhD

This material is based upon work supported by

Outline

- ➔ Intro to Cheating in Video Games
- ➔ Motivations & Challenges
- ➔ Our Contribution
- ➔ A Brief Overview of Machine Learning
- ➔ Proposed Framework Details
- ➔ Empirical Evaluation
- ➔ Future Works

Video Game Industry

- One of the largest Entertainment Industries
- Global revenue is expected to reach nearly $180 billion in 2020 [1]
- Most revenue comes from
  - In-game purchases
  - Advertisements
  - Consoles and controllers

- A serious impediment damaging this multi-billion dollar industry: **Cheating**

[1] https://www.marketwatch.com/story/videogames-are-a-bigger-industry-than-sports-and-movies-combined-thanks-to-the-pandemic-11608654990

What is Cheating in Video Games?

- Any behavior performed by a game player to change normal execution of game-play and obtain unfair advantages while playing video games

- The game player who cheats is called a <u>Cheater</u>

How Gamers Cheat: Techniques

- Using Cheat Code
- Modifying Game Code
- Modifying System Software
- Modifying Game Traffic
- Using Game Bots

Example of using cheat codes in Counter-Strike game

How Gamers Cheat: Resources

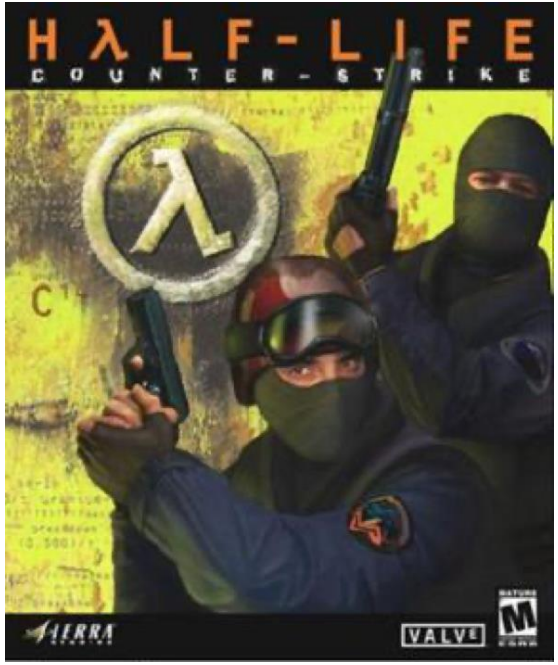
- Gaming community
- Social media [2]
  - Discord
  - Instagram

[2] https://egmnow.com/the-human-side-of-those-who-cheat-at-and-hack-games/

Online Video Game Example

- Counter-Strike 1.6 [Ɣ]
  - Multiplayer first-person shooting game
  - One of the most popular online games
  - Many cheats available

- Some cheats
  - Wall-hack
  - Speed-hack
  - Aim-bot
  - Trigger-bot
  - Artificial-lag



[Ɣ] https://www.valvesoftware.com/en/

Why Gamers Cheat?

Profit

Competitiveness

Entertainment

Damages of Cheating

- Adversely affects game's popularity and reputation
  - 77% of players are likely to stop playing online multiplayer games if they suspect other players are cheating [3]
  - 60% have had negative gaming experiences because of cheating [3]

- Hurts revenue
  - 48% players would be reluctant to purchase any in-game content if other players are cheating [3]

source: vecteezy.com

[3] https://resources.irdeto.com/irdeto-global-gaming-survey/infographic-cheating-game-over

Outline

- ➔ Intro to Cheating in Video Games
- ➔ Motivations, Challenges, and Our Contribution
- ➔ A Brief Overview of Machine Learning
- ➔ Proposed Framework
- ➔ Empirical Evaluation
- ➔ Future Works

Motivation

- Resist cheating trend in online games
- Limited client-side information
  - Detecting cheats is challenging mainly due to the limited client-side information.
- Complexity
  - The cheating techniques are unknown and complex.

Challenges

- Game dependent:
  - Most cheat detection methods analyze decrypted game-dependent data.
- Covariate shift:
  - The assumption of training set and test set having similar distribution may not hold.
  - This may be due to sampling bias caused by label scarcity, inaccessible, and the cost of label procurement.
- Limited labeled data:
  - Supervised learning models such as SVM, kNN, and neural network typically perform well when training and test datasets have similar distribution.
  - Supervised learning mechanism not suitable for very limited training data.
- Computational efficiency:
  - Current cheat detection methods mainly have delayed detection.
  - A large delay in detection (e.g., using game logs etc.) may not be effective to act upon cheaters at the right time.

Contribution

- Game independence:
  - In this work, we analyze the game traffic, which is encrypted and game independent.
  - It is easier to evaluate over encrypted traffic since most games are not open-source.
- Covariate shift:
  - We utilize relative density ratio to estimate importance weights associated with training data instances.
- Scalability:
  - For server-side cheat detection deployment, we demonstrate the scalability of our proposed approach using Apache Spark and Graphics Processing Unit (GPU).

Outline

- ➔ Intro to Cheating in Video Games
- ➔ Motivations, Challenges, and Our Contribution
- ➔ A Brief Overview of Machine Learning
- ➔ Proposed Framework
- ➔ Empirical Evaluation
- ➔ Future Works

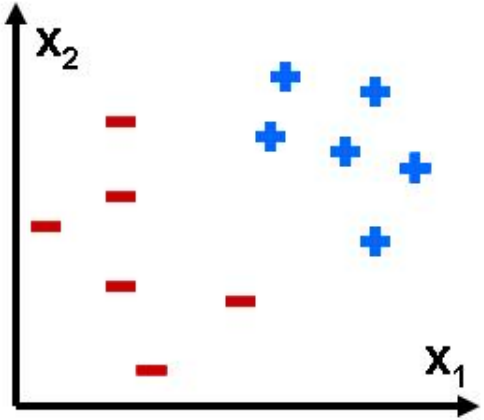
What is Data Classification

- Classification problem can be described as:

  Given a training data TD = $\{(x_1, y_1), (x_2, y_2), \ldots\ldots, (x_N, y_N)\}$.
  Design a function $f: X \rightarrow Y$, that maps any observed data x to a certain class y.

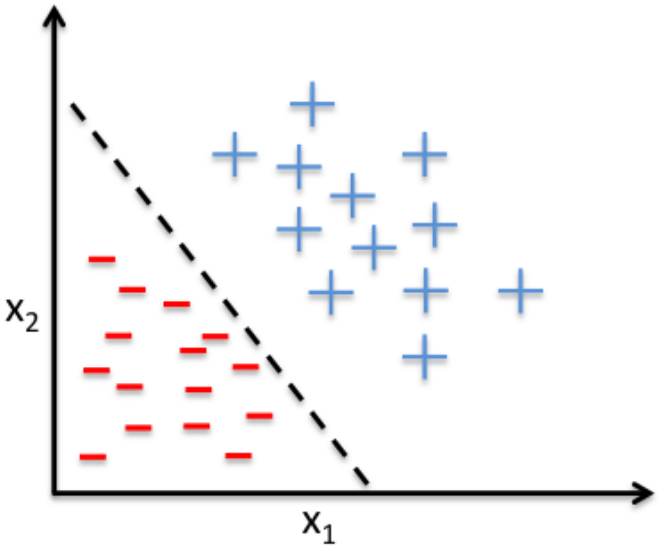
Binary Classification

- Is a classification problem, where we have two classes (we often call one class positive and the other negative)



https://alliance.seas.upenn.edu/~cis520/dynamic/2017/wiki/index.php?n=Lectures.Classification

Binary Classification (linearly separable data)



http://sebastianraschka.com/Articles/2015_singlelayer_neurons.html

Binary Classification (linearly separable data)

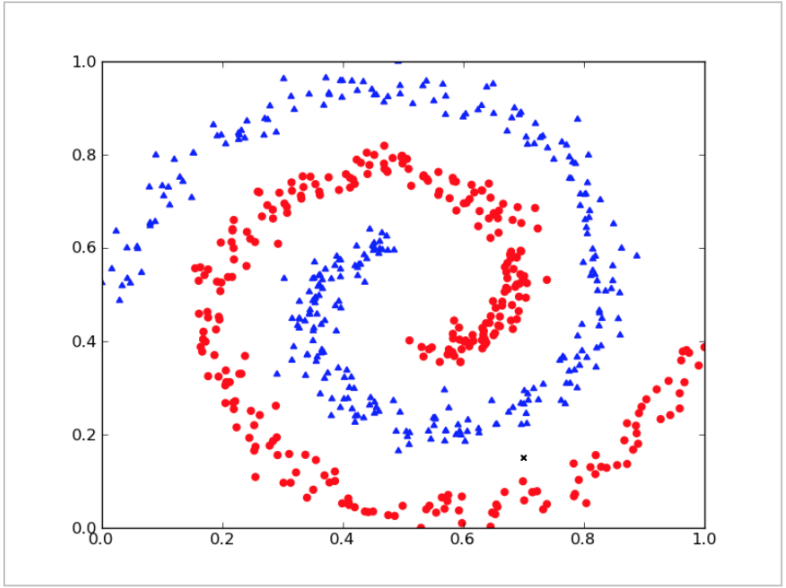
- Our goal is to find a hyperplane such that

  $Y^i = sign(w^T x^i + b)$, for all $(x^i, y^i) \in$ Training data

- We predict the class y' of data item x' as

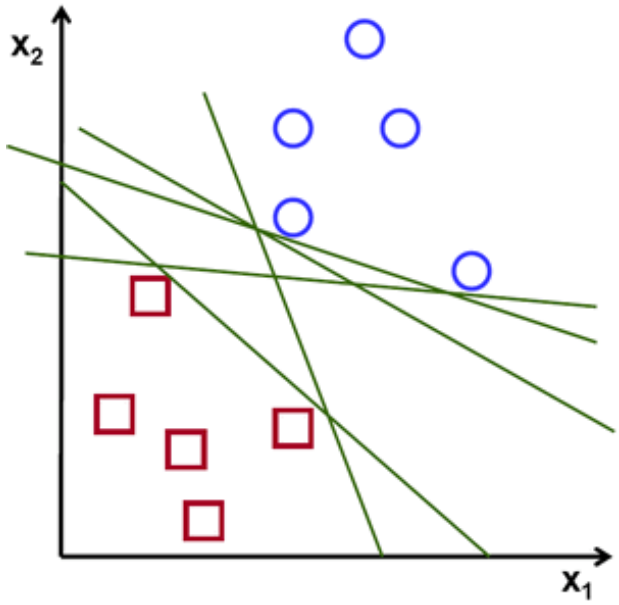
  $Y' = sign(w^T x' + b)$

Binary Classification (linearly inseparable data)



https://www.classes.cs.uchicago.edu/archive/2013/winter/12200-1/assignments/pa4/index.html

What is the best linear Separator?



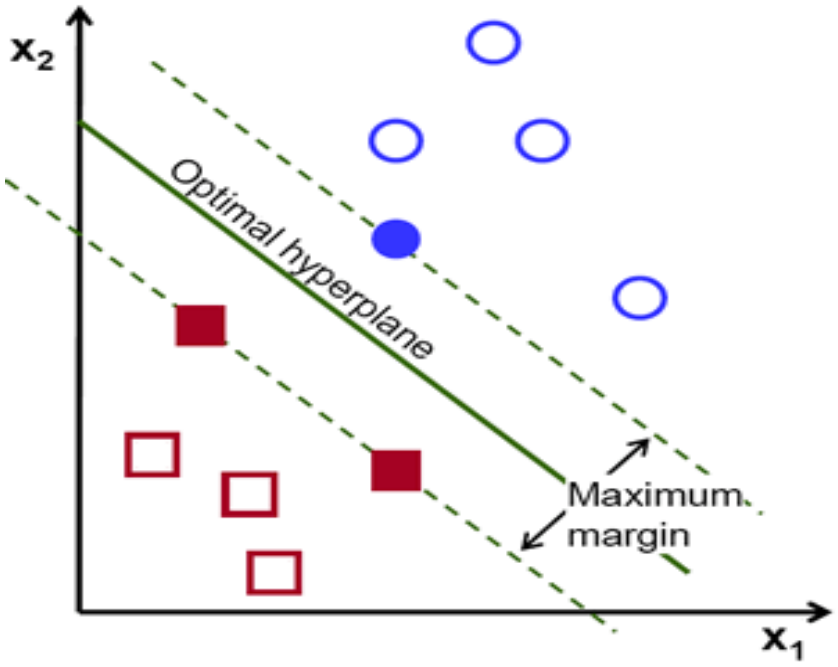
https://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html

Support vector machines (SVMs)

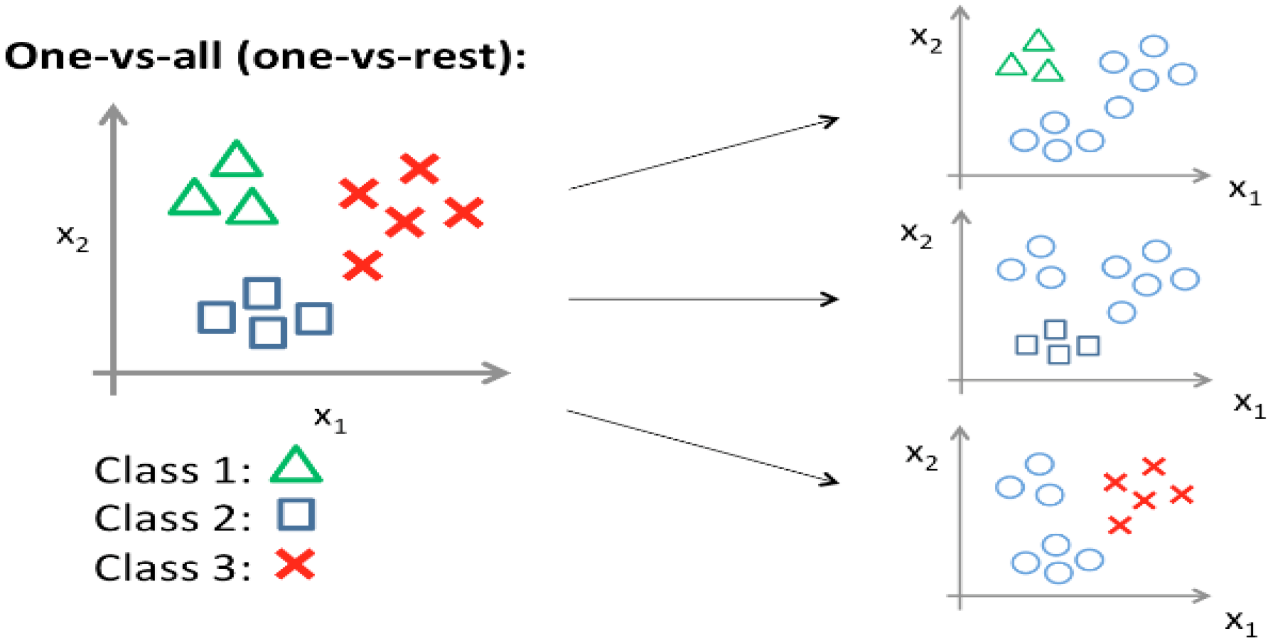
Define the margin to be the twice the distance of the closest data point to the classifier

SVM chooses the classifier (hyperplane) that maximize the margin: Good according to intuition, theory, practice.

SVMs



SVMs- Multi-class classification



https://www.linkedin.com/pulse/multi-class-classification-imbalanced-data-using-random-burak-ozen/

SVMs- one-vs-one

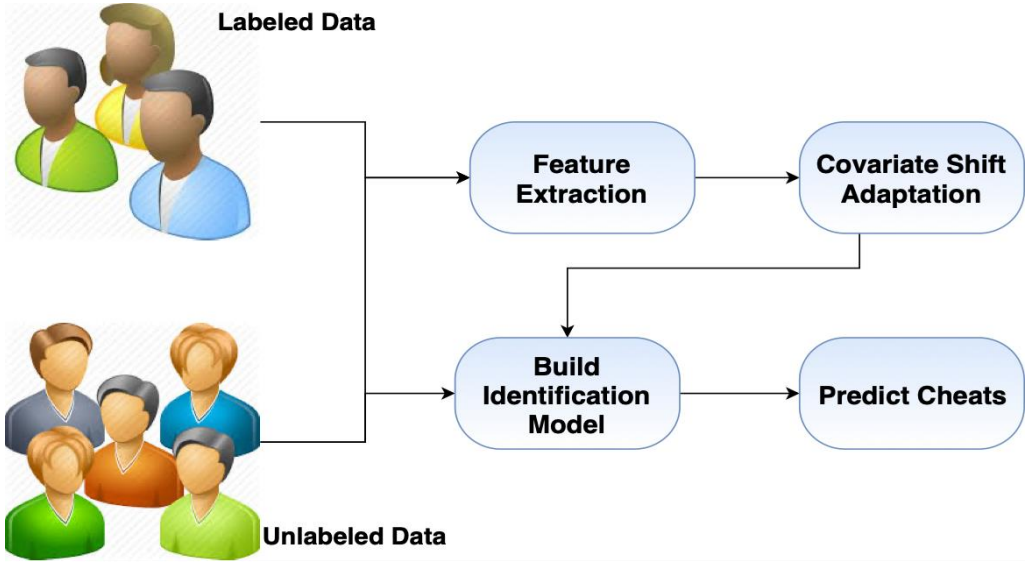
Alternatively we can construct a classifier for all possible pairs of labels.

Given a new data point, we can classify it by majority vote.

Outline

- ➔ Intro to Cheating in Video Games
- ➔ Motivations, Challenges, and Our Contribution
- ➔ A Brief Overview of Machine Learning
- ➔ Proposed Framework
- ➔ Empirical Evaluation
- ➔ Future Works

Overview of GCI framework



Feature Extraction

- Packets are encrypted.
- Extract features from packet headers.

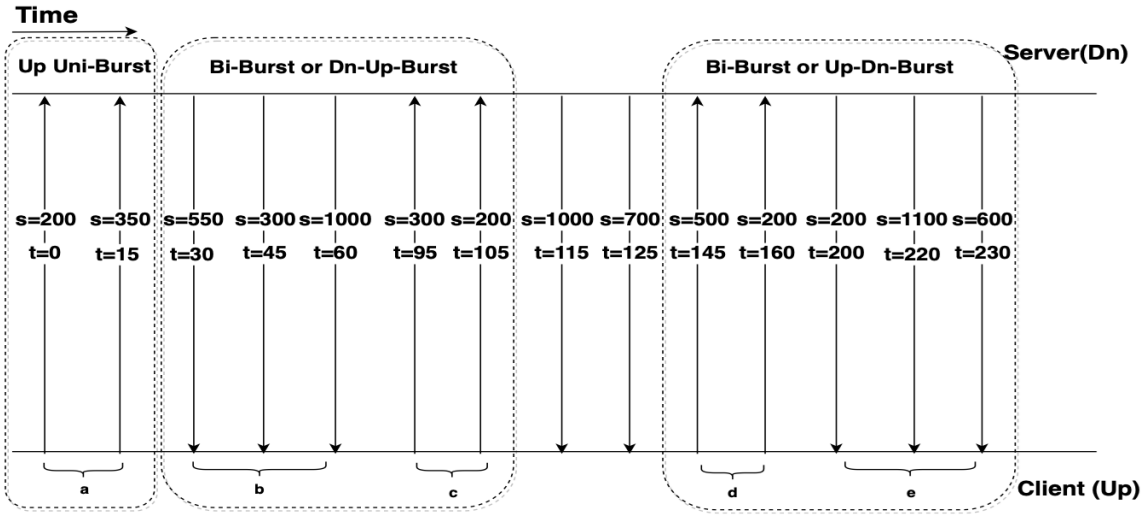
- Some general features:
  - Number of incoming packets.
  - Number of outgoing packets.
  - Sum of incoming packet sizes.
  - Sum of outgoing packet sizes.

Feature Extraction: BIND

- **BIND** (Fingerprinting with BI-directioNal Dependence) [4][5]:
  - Works with Bursts
  - A burst is a sequence of consecutive packets transmitted along the same direction
  - Uni-Burst:
    - Size
    - Time
    - Direction
    - Number of packets in the burst
  - Bi-Burst:
    - Size
    - Time
    - Number of packets in the burst

[4] K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham, “Adaptive encrypted traffic fingerprinting with bi-directional dependence,” in Proceedings of the 32Nd Annual Conference on Computer Security Applications, ser. ACSAC ’16. Los Angeles, California, USA, 2016, pp. 177–188.
[5] Al-Naami, K., El Ghamry, A., Islam, M.S., Khan, L., Thuraisingham, B.M., Hamlen, K.W., Alrahmawy, M. and Rashad, M., 2019. Bimorphing: A bi-directional bursting defense against website fingerprinting attacks. *IEEE Transactions on Dependable and Secure Computing*.

Feature Extraction: BIND



DnUp-1850-500 UpDn-700-1900

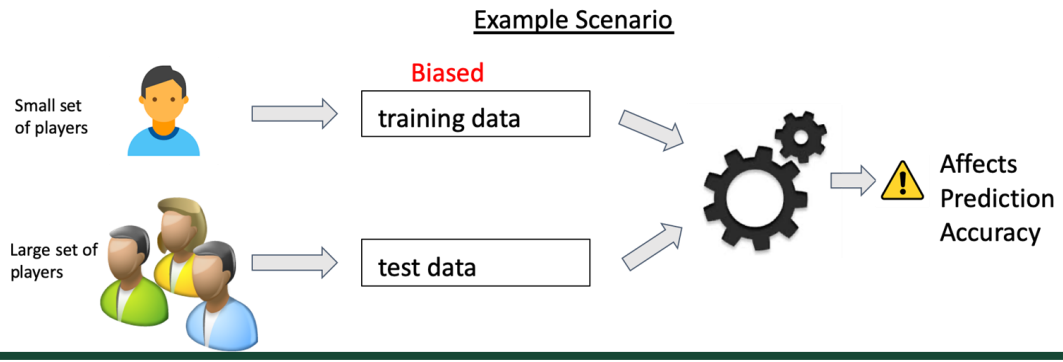
An example of feature extraction procedure following BIND

- All the features are concatenated to form a large array of features (histograms) to be extracted from each trace. A set of multiple traces represented in this manner forms the training and testing set.

Covariate Shift Problem

- What if we do not find a good training set?
- Different sets of players may cause biased training data with respect to test data.

- **Solution:**
  - We utilize relative density ratio to estimate importance weights associated with training data instances.
  - We propose a expectation-maximization technique to automatically learn model parameters for relative density ratio estimation from available data.



Covariate Shift Adaptation: Spark Implementation

- Scalability:
  - As our proposed work contains a great deal of large-scale matrix multiplications, we utilize Spark to accelerate the process.
  - We separate the large matrix computation into small blocks and distribute the small tasks parallel on Spark clusters.

- Computation efficiency:
  - Applying Spark reduces the execution time and improves performance when we have large data set.

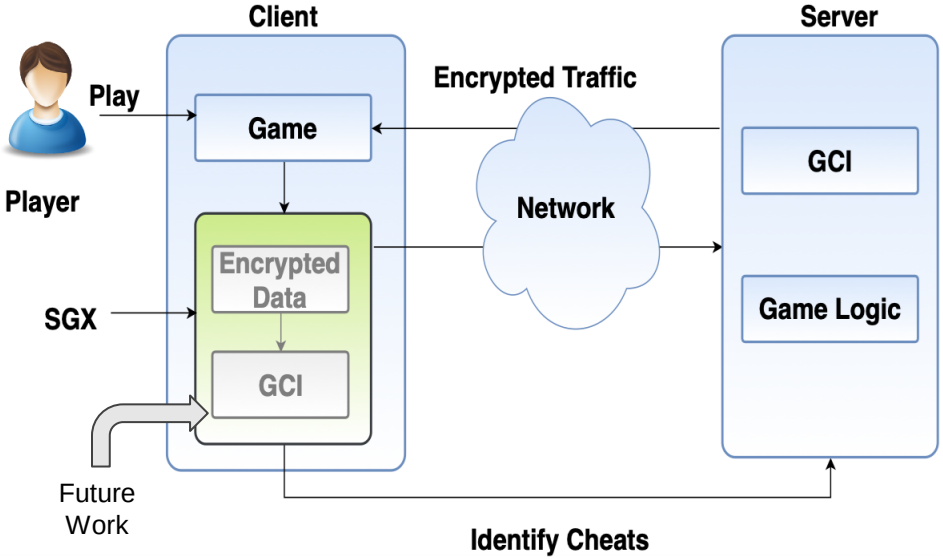
Covariate Shift Adaptation: GPU Implementation

- Graphics Processing Units (GPU)
  - Powerful parallel processing capability with abundant computing cores
  - High memory bandwidth
  - Reduces processing burden from the CPU

- We use GPU to accelerate major time-consuming operations
  - Learning parameters for relative density ratio
  - Hyper-parameters searching for the estimator.

Deployment

- ❑ Deploy GCI framework in game server-side
- ❑ Since our mechanism is not game-specific, we can deploy cheat detection on the client-side as well. **(Future Work)**
- ❑ We plan to deploy our GCI framework in SGX [6] in game client-side for future work.



[6] V. Costan and S. Devadas, “Intel sgx explained.” IACR Cryptology ePrint Archive, vol. 2016, no. 086, pp. 1–118, 2016.

Outline

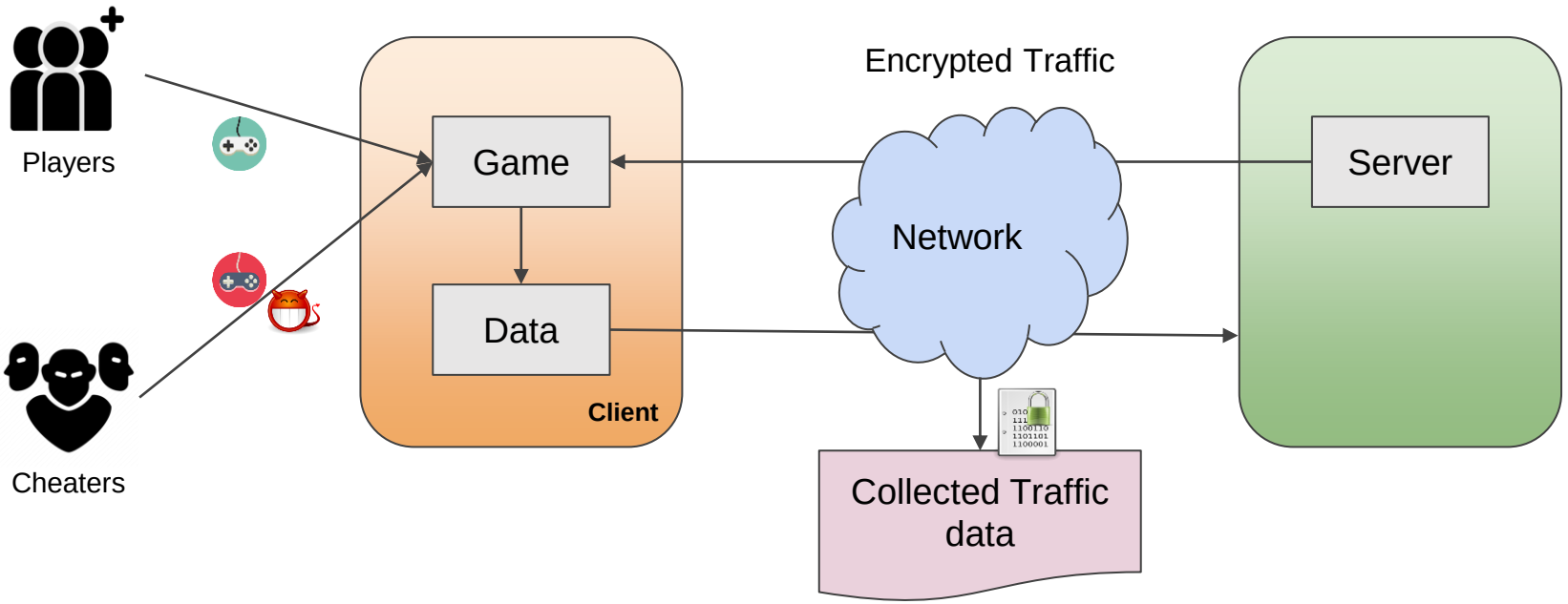
- ➔ Intro to Cheating in Video Games
- ➔ Motivations, Challenges, and Our Contribution
- ➔ A Brief Overview of Machine Learning
- ➔ Proposed Framework
- ➔ Empirical Evaluation
- ➔ Future Works

Empirical Evaluation: Data Collection

- We collect game traffic with help of students from class CS 6301: Cyber Security Essentials of University of Texas at Dallas and Big Data Analytics and Management Lab.
- In total 20 students participate to collect data.
- Students install in their personal machines the game Counter-Strike 1.6 and the three selected cheat types downloaded from a diverse community of popular cheating sources.[1,2]
- They connect to the server and play the game in both normal game mode as well as using the cheats applied to the game.

[1] https://www.gamespot.com/counter-strike/cheats/
[2] https://www.unknowncheats.me/forum/index.php

Empirical Evaluation: Data Collection



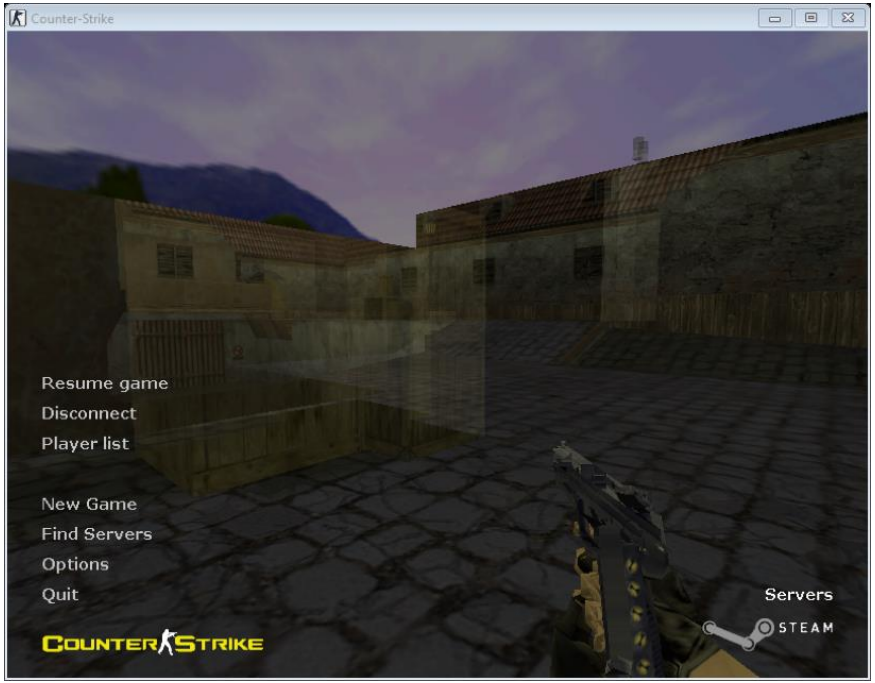
Empirical Evaluation: Counter-strike Cheats

- Aim-bot
  - Enables automatic targeting the opponent while shooting.
  - This targeting works even if the opponent is too far away or behind walls.

- Speed-hack
  - Enables speed increase in player's movement while playing the game.
  - A player can apply different variations of speeds and play the game.

- Wall-hack
  - Makes the walls transparent for the player so that he or she can see the enemy through the walls.

Wall-hack Example



Empirical Evaluation: Experiment Settings

- Feature extraction:
  - We first extract features following [4][5]

- Generate training and test data:
  - We generate data in different 10 groups by selecting different fixed sized training set and run experiment by cross-validation.

[4] K. Al-Naami, S. Chandra, A. Mustafa, L. Khan, Z. Lin, K. Hamlen, and B. Thuraisingham, “Adaptive encrypted traffic fingerprinting with bi-directional dependence,” in *Proceedings of the 32Nd Annual Conference on Computer Security Applications*, ser. ACSAC ’16. Los Angeles, California, USA, 2016, pp. 177–188.

Empirical Evaluation: Experiment Settings

- Multi class labels:
  - Aim-bot
  - Speed-hack
  - Wall-hack
  - Normal (without cheats)

- Binary class labels:
  - Cheats (aim-bot, speed-hack, wall-hack)
  - Normal (without cheats)

Empirical Evaluation: Baseline Methods

Baseline Methods	Description
KMSVM	Equip KMM[7] with base classifier weighted SVM to build classification models.
KLISVM	Equip KLIEP[8] with base classifier weighted SVM to build classification models.
SVM	Multi class Support Vector Machine.
Proposed Method	**Description**
GCI	Equip revised RULSIF with base classifier weighted SVM to build classification models

[7] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Scholkopf, “Correcting sample selection bias by unlabeled data,” in Proceedings of the 19th International Conference on Neural Information Processing Systems, ser. NIPS’06. Cambridge, MA, USA: MIT Press, 2006, pp. 601–608.
[8] Y. Kawahara and M. Sugiyama, “Sequential change-point detection based on direct density-ratio estimation,” Stat. Anal. Data Min., vol. 5, no. 2, pp. 114–127, Apr. 2012

Empirical Evaluation: Performance

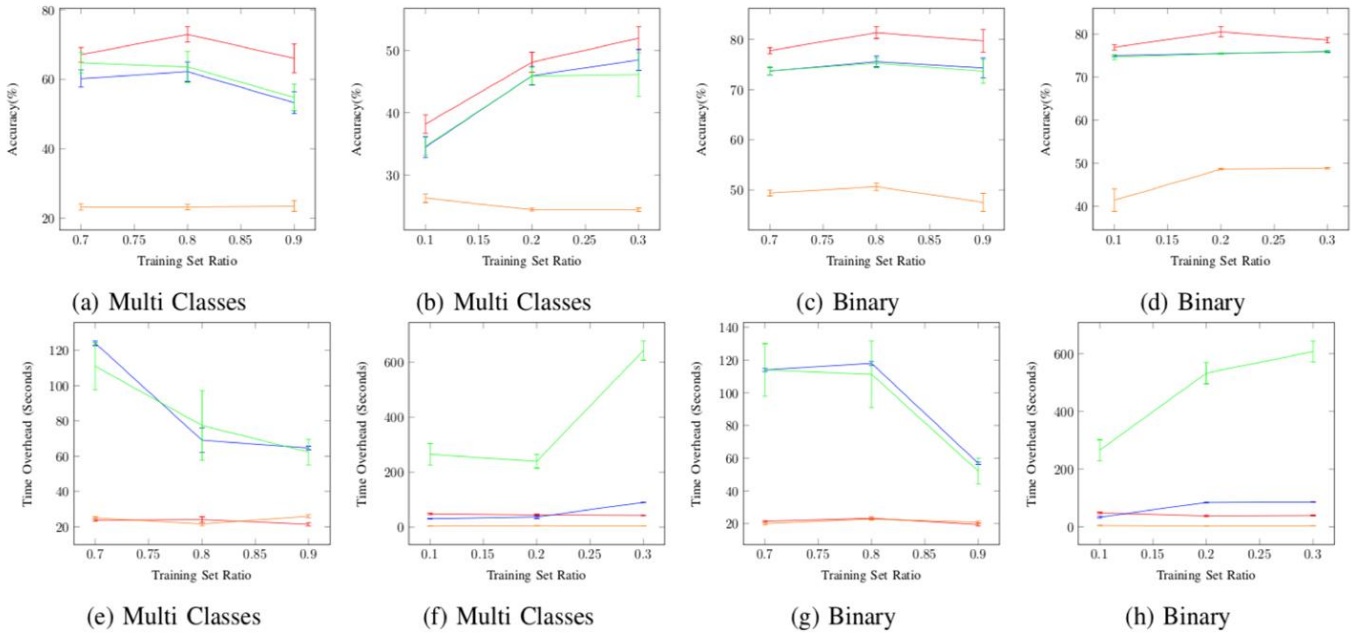
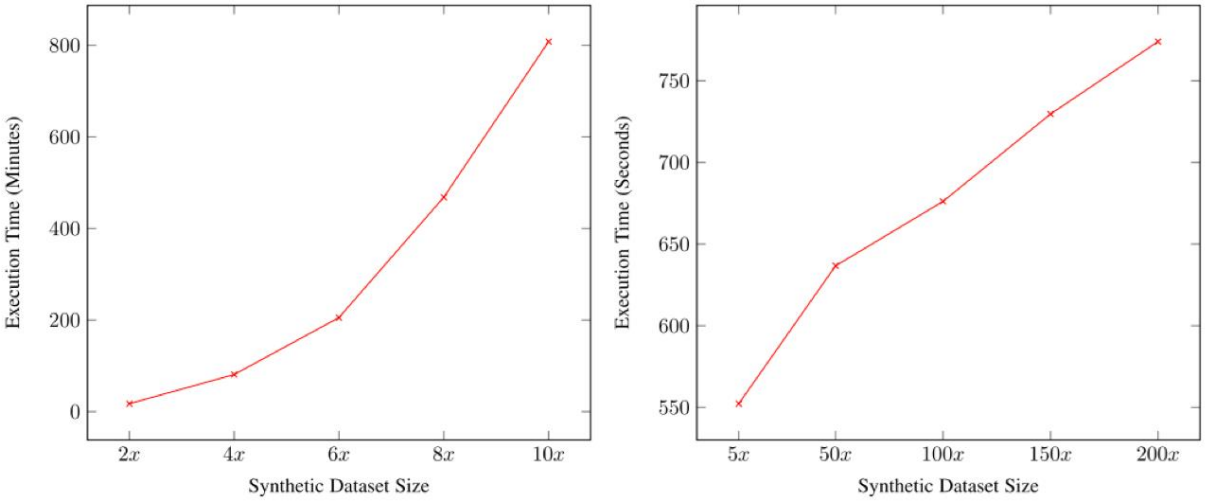


Figure 5: Performance of classification for all approaches. (GCI; KMSVM; KLISVM; SVM).

Empirical Evaluation:Performance



(a) Spark (b) GPU

Fig. 6. Performance of Spark and GPU for large datasets.

Outline

- ➔ Intro to Cheating in Video Games
- ➔ Motivations, Challenges, and Our Contribution
- ➔ A Brief Overview of Machine Learning
- ➔ Proposed Framework
- ➔ Empirical Evaluation
- ➔ Future Works

Future Direction

- We plan to investigate the performance of GCI when more cheating techniques are introduced.
- We will consider other games and examine how GCI performs.
- We plan to perform secure execution of cheat detection at the client-side with Trusted Execution Environments such as Intel SGX platform.
- We will explore similar detection methods for distributed massive online games, i.e., those which do not have a server-client architecture.

Thank you

Contact information:
lkhan@utdallas.edu
md.shihabul.islam@utdallas.edu