You are a Game Bot:
Uncovering Game Bots in MMORPGs via Self-similarity in the Wild

Eunjo Lee (NCSoft)
Jiyoung Woo (Korea University)
Hyoungshick Kim (Sungkyunkwan University)
Aziz Mohaisen (State University of New York at Buffalo)
Huy Kang Kim (Korea University)
Contents

Introduction
Feature selection and modeling
Experiments
Model maintenance
Real-World deployment
Conclusions
Introduction
Introduction

Game BOT

- Program that plays a game autonomously (instead of human users)
Introduction

Real Money Trading (RMT)

- Collect valuable items and monetize it by trading item to others
Introduction

Gold Farming Group (GFG)
Introduction

Game BOT

https://www.youtube.com/watch?v=k6tk8_R2w08
Introduction

Game BOT

- Widespread cheating in online games
- Collapse of an in-game economy
- Cause a human users’ churn
- Reduce the revenue
Introduction

Countermeasures

• Client-side
  • Bot process detection using anti-malware programs
• Server-side
  • Bot classification using game log analysis
Introduction

Machine Learning-based Approach

Game Logs

<table>
<thead>
<tr>
<th>Character ID</th>
<th>T1</th>
<th>T2</th>
<th>T3</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>686042</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>854209</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1032131</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1049483</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1340479</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1352850</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1771815</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1832497</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1884884</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2130576</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2445903</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Ground Truth

Learning Algorithm

Prediction Model
Introduction

Challenges

Raw data collection
Preprocessing
Cleaning
Visualizing
...

Game Bots' Pattern
Introduction

Challenges

Game A Bots’ Pattern

Game B Bots’ Pattern

High cost
time consuming

Raw data collection

Preprocessing

Cleaning

Visualizing

...
Introduction

Challenges

Game update
Bot change

Consistent maintenance
Introduction

Our proposals

• Using self-similarity as a generic feature
  • Focus on the repetitive activities of game bots, not specific behavior
• Proposing framework to maintain a prediction model autonomously
  • Detect the change in performance of the prediction model and retrain it
Feature Selection and Modeling
Self-similarity

Definition

• Measurement of the similarity of periodic actions per user
Self-similarity

Motivation and consideration

• Intrinsic attributes
  • Bot programs repeat routines using predetermined settings
  • Human users may exhibit similar behavior, but not for long period of time

• Stability
  • Little effect of game update or bot program changes
  • Considering various actions rather than a single action

• Computing efficiency
  • Easy to apply distributed algorithms (i.e. MapReduce) for log processing
Self-similarity

Detailed process

• Generating log vectors
• Measuring cosine similarity
• Measuring self-similarity
Self-similarity

Generating log vectors

<table>
<thead>
<tr>
<th>time</th>
<th>Event id</th>
<th>Character info.</th>
<th>Other info.</th>
</tr>
</thead>
<tbody>
<tr>
<td>15/08/13 12:00:12.131</td>
<td>1205</td>
<td>AAA, 34</td>
<td>N/A</td>
</tr>
<tr>
<td>15/08/13 12:00:14.237</td>
<td>1204</td>
<td>AAA, 34</td>
<td>N/A</td>
</tr>
<tr>
<td>15/08/13 12:00:59.436</td>
<td>1208</td>
<td>AAA, 34</td>
<td>Ogre</td>
</tr>
<tr>
<td>15/08/13 12:00:59.436</td>
<td>1208</td>
<td>AAA, 34</td>
<td>Ogre</td>
</tr>
<tr>
<td>15/08/13 12:00:59.857</td>
<td>1208</td>
<td>AAA, 34</td>
<td>Troll</td>
</tr>
<tr>
<td>15/08/13 12:01:17.019</td>
<td>1022</td>
<td>AAA, 34</td>
<td>Ring</td>
</tr>
<tr>
<td>15/08/13 12:01:21.341</td>
<td>1022</td>
<td>AAA, 34</td>
<td>Sword</td>
</tr>
<tr>
<td>15/08/13 12:01:23.151</td>
<td>1205</td>
<td>AAA, 34</td>
<td>N/A</td>
</tr>
<tr>
<td>15/08/13 12:01:54.354</td>
<td>1204</td>
<td>AAA, 34</td>
<td>N/A</td>
</tr>
<tr>
<td>15/08/13 12:01:56.445</td>
<td>1208</td>
<td>AAA, 34</td>
<td>Wolf</td>
</tr>
<tr>
<td>15/08/13 12:02:07.351</td>
<td>1205</td>
<td>AAA, 34</td>
<td>N/A</td>
</tr>
<tr>
<td>15/08/13 12:02:41.847</td>
<td>1204</td>
<td>AAA, 34</td>
<td>N/A</td>
</tr>
<tr>
<td>15/08/13 12:02:47.650</td>
<td>1208</td>
<td>AAA, 34</td>
<td>Ogre</td>
</tr>
<tr>
<td>15/08/13 12:03:09.353</td>
<td>1208</td>
<td>AAA, 34</td>
<td>Ogre</td>
</tr>
</tbody>
</table>

Time period (hour:min) | Log count per event id
--- | ---
12:00 | 0 1 1 3
12:01 | 2 1 1 1
12:02 | 0 1 1 1
12:03 | 0 0 0 1
Self-similarity

Measuring the cosine similarity between log vector($V_t$) and unit vector($E$)

\[
\cos(\theta) = \frac{A \cdot B}{||A|| ||B||} = \frac{\sum A_i \times B_i}{\sqrt{\sum (A_i)^2} \times \sqrt{\sum (B_i)^2}}
\]

\[
= \frac{(2 \times 1 + 1 \times 1)}{\sqrt{2 \times 2 + 1 \times 1} \times \sqrt{1 \times 1 + 1 \times 1}}
\]

\[
= \frac{3}{\sqrt{5} \times \sqrt{2}}
\]

\[\approx 0.948\]
Self-similarity

Measuring self-similarity

• Measuring std. of cosine similarity and transforming using the following model

  \[ H = 1 - \frac{1}{2} \sigma, \quad (0.5 \leq H \leq 1, \sigma: \text{std. deviation of cosine similarity}) \]
Modeling and Evaluation

Modeling

• Logistic regression
  • Calculating the probability of a character being a game bot
Experiments
## Experiments

### Datasets

<table>
<thead>
<tr>
<th></th>
<th>Lineage</th>
<th>Aion</th>
<th>B&amp;S</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Release year</strong></td>
<td>1997</td>
<td>2008</td>
<td>2012</td>
</tr>
<tr>
<td><strong>Daily active users</strong></td>
<td>300K</td>
<td>200K</td>
<td>100K</td>
</tr>
<tr>
<td><strong>Concurrent users</strong></td>
<td>150K</td>
<td>80K</td>
<td>50K</td>
</tr>
</tbody>
</table>
Experiments

Cosine similarities

- Bots have cosine similarities with fewer variations than human users
Experiments

Self-similarity

- Almost bots have higher values than human users
Feature selection

Additional feature selection

- Exceptional cases – short time playing or no activities over long time
- Outliers

<table>
<thead>
<tr>
<th>No.</th>
<th>Field name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>self_sim</td>
<td>Self-similarity</td>
</tr>
<tr>
<td>2</td>
<td>cosim_count</td>
<td>Count of a set of log vectors</td>
</tr>
<tr>
<td>3</td>
<td>cosim_uniq_count</td>
<td>Unique count of a set of log vectors</td>
</tr>
<tr>
<td>4</td>
<td>cosim_zero_count</td>
<td>Count of data in which cosine similarity is zero</td>
</tr>
<tr>
<td>5</td>
<td>cosim_mode</td>
<td>Count of data that appears most often in a set of log vectors</td>
</tr>
<tr>
<td>6</td>
<td>total_log_count</td>
<td>Total count of logs generated by user</td>
</tr>
<tr>
<td>7</td>
<td>main_char_level</td>
<td>Character level</td>
</tr>
<tr>
<td>8</td>
<td>total_use_time_min</td>
<td>Play time during certain period per user</td>
</tr>
<tr>
<td>9</td>
<td>npc_kill_count</td>
<td>NPC kill count</td>
</tr>
<tr>
<td>10</td>
<td>trade_get_count</td>
<td>Count of trade in which user takes item</td>
</tr>
<tr>
<td>11</td>
<td>trade_give_count</td>
<td>Count of trade in which user gives items</td>
</tr>
<tr>
<td>12</td>
<td>retrieve_count</td>
<td>Count of activity in which user retrieve items from warehouse</td>
</tr>
<tr>
<td>13</td>
<td>deposit_count</td>
<td>Count of activity in which user deposits items to warehouse</td>
</tr>
<tr>
<td>14</td>
<td>log_count_per_min</td>
<td>Average count of logs are generated per minute</td>
</tr>
</tbody>
</table>
Performance evaluation

- Model1: using only self-similarity. Model2: using all features

<table>
<thead>
<tr>
<th>Game</th>
<th>BOT</th>
<th>Human</th>
<th>AUC (model 1)</th>
<th>AUC (model 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lineage</td>
<td>128</td>
<td>149</td>
<td>0.8967</td>
<td>0.9455</td>
</tr>
<tr>
<td>Aion</td>
<td>186</td>
<td>160</td>
<td>0.9557</td>
<td>0.9942</td>
</tr>
<tr>
<td>B&amp;S</td>
<td>131</td>
<td>129</td>
<td>0.8280</td>
<td>0.9399</td>
</tr>
</tbody>
</table>
Model maintenance

Motivation and consideration

• How to optimize the time for retraining
  • Too often -> high cost
  • Too rare -> obsolete model
• How to retrain a model autonomously
Model maintenance

System Flow

Inspector

Ground Truth

Modeler

Model (PMML)

Predictor

Game Logs

Change Detector

BOT Detection System

Model maintenance

System Flow

Inspector

Ground Truth

Modeler

Model (PMML)

Predictor

Game Logs

Change Detector

BOT Detection System
Model maintenance

Logic Flow

- If change is detected, retraining the model
- Notifying to operator, if new model is invalid or change is detected consecutively

- Calculate bot probability
- **Change detected?**
  - yes
  - no → End

- **Already retraining?**
  - yes
  - no → Model retraining
    - invalid → Validation check
    - valid
      - End
    - End

- Notify to operator
Model maintenance

Logic Flow

• If change is detected, retraining the model
• Notifying to operator, if new model is invalid or change is detected consecutively

 EWMA Algorithm

Logic Flow

- Calculate bot probability
- Change detected?
  - yes
  - yes
  - no
  - no
- End
- Already retraining?
  - yes
  - Notify to operator
  - End
  - Model retraining
    - invalid
    - Validation check
    - valid
Model maintenance

EWMA algorithm

- Calculating the correlation coefficient of bot probability between time $t$ and $t-1$

<table>
<thead>
<tr>
<th>User</th>
<th>Bot probability (time $t$)</th>
<th>Bot probability (time $t-1$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.99</td>
<td>0.95</td>
</tr>
<tr>
<td>B</td>
<td>0.95</td>
<td>0.92</td>
</tr>
<tr>
<td>C</td>
<td>0.23</td>
<td>0.25</td>
</tr>
<tr>
<td>D</td>
<td>0.55</td>
<td>0.55</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Model maintenance

EWMA algorithm

• Calculating the correlation coefficient of bot probability between time t and t-1

• Calculating the weighted moving average of coefficients

(X: coefficient, Z: moving average)
Model maintenance

EWMA algorithm

• Calculating the correlation coefficient of bot probability between time t and t-1

• Calculating the weighted moving average of coefficients

• Measuring upper and lower control limits
Model maintenance

EWMA algorithm

• Calculating the correlation coefficient of bot probability between time t and t-1

• Calculating the weighted moving average of coefficients

• Measuring upper and lower control limits

• Retraining the model, unless \( LCL < Z_t < UCL \)
Real-World Deployment
Real-World Deployment

BOT detection system – dashboard

- Provide the trend of numbers or rates of BOT, and the chart of BOT statistics by main activity zone
Real-World Deployment

BOT detection system – search and filter

- Search and filter the list of accounts to ban

Fill in the conditions to filter accounts to ban

Print the list of accounts to ban according to the search conditions
Conclusion
Conclusion

Contributions

• We proposed self-similarity as a feature and demonstrated its effectiveness with real datasets

• We proposed a bot detection framework that includes a detection model maintenance process

• We implemented the proposed framework and utilized it for live MMORPGs
Conclusion

Future works – short-time playing bot

- Undetected massive number of bots playing for less than 10 hours per week
- Star-shaped trading network structure
Conclusion

Future works – occasional bot users

- Human players playing for hours and then turning on a bot for a few hours
- Self-similarities have pulse, if we use short period of time for aggregation
Questions and Answers