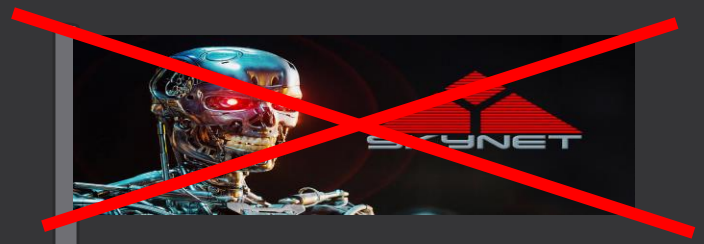


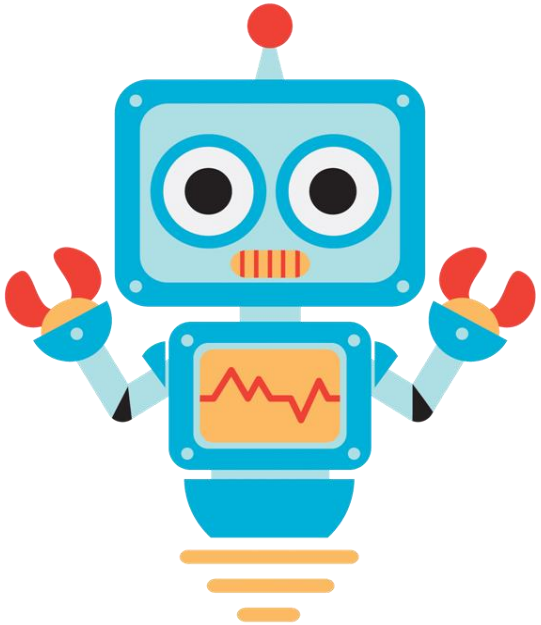
BotDetect

Team 8 Demo 2: Data Exploration

Carlos Acuna, Corbin Graham, Jose Medina Mani, Adam Riffel, Nhan Tran

Topic

- Primary Research Question
  - Is it possible to differentiate between a human and bot playing a game?
- Secondary Research Questions
  - Can we do this from a naive approach?
  - Can we identify a human or bot in real time?
  - Can we expand this to identifying individual users?
  - Can our solution be generalized to multiple platforms?

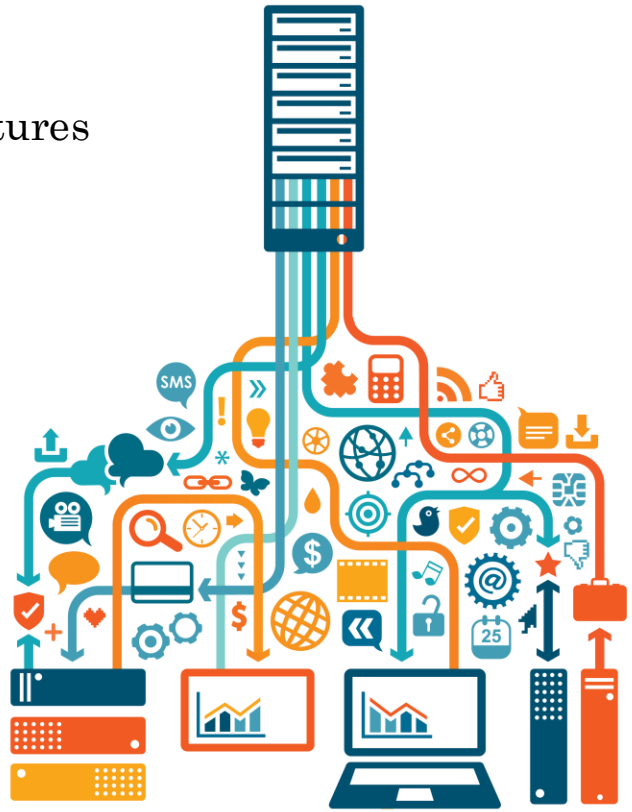


Approach Questions

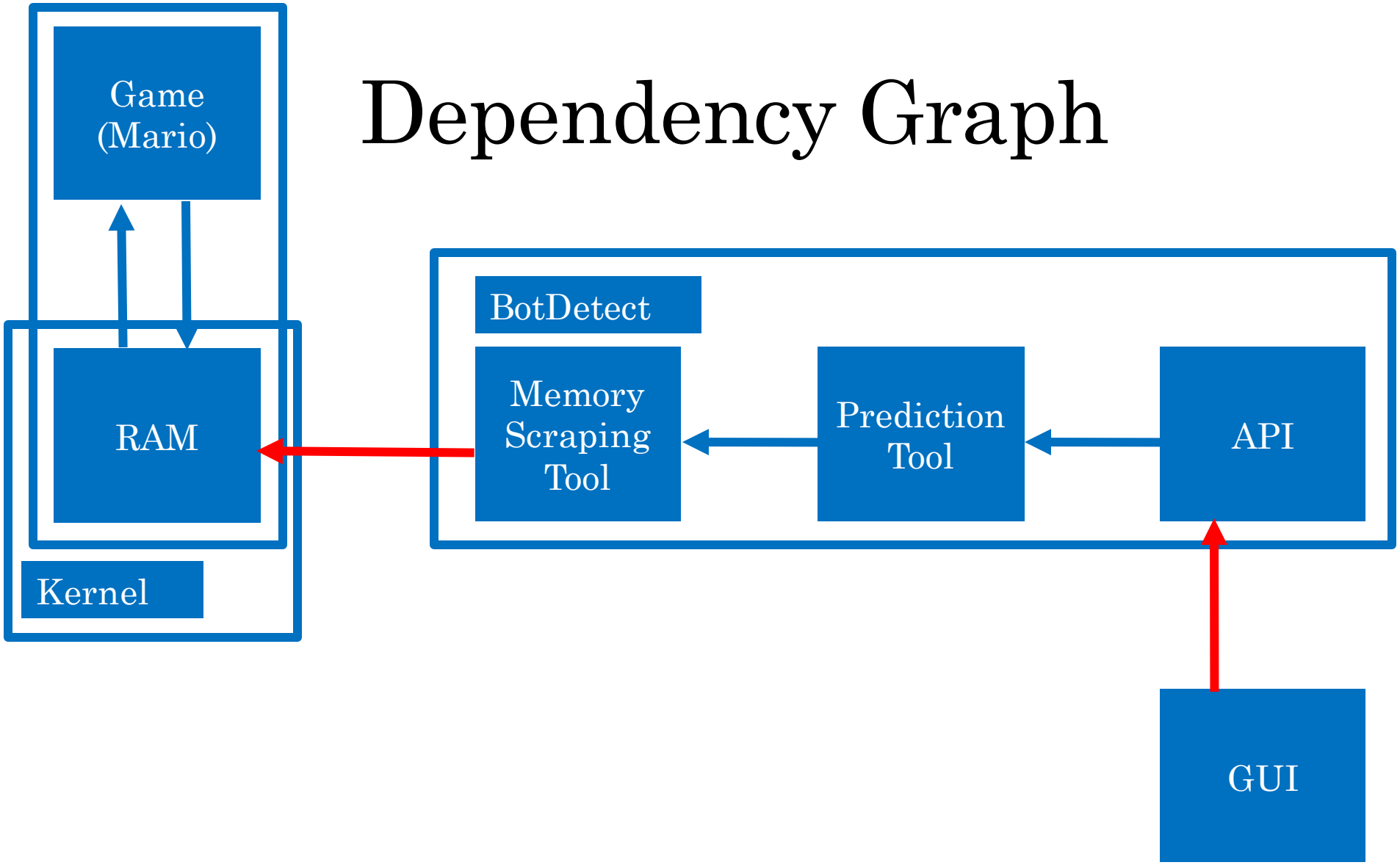
- Data Gathering
  - How will we gather the data?
- Data Cleaning
  - Given the data, what needs to be cleaned?
- Evaluation
  - How can we evaluate the data effectively to predict our identifier?
- Implementation
  - Can we implement this for real-time prediction?

Approach

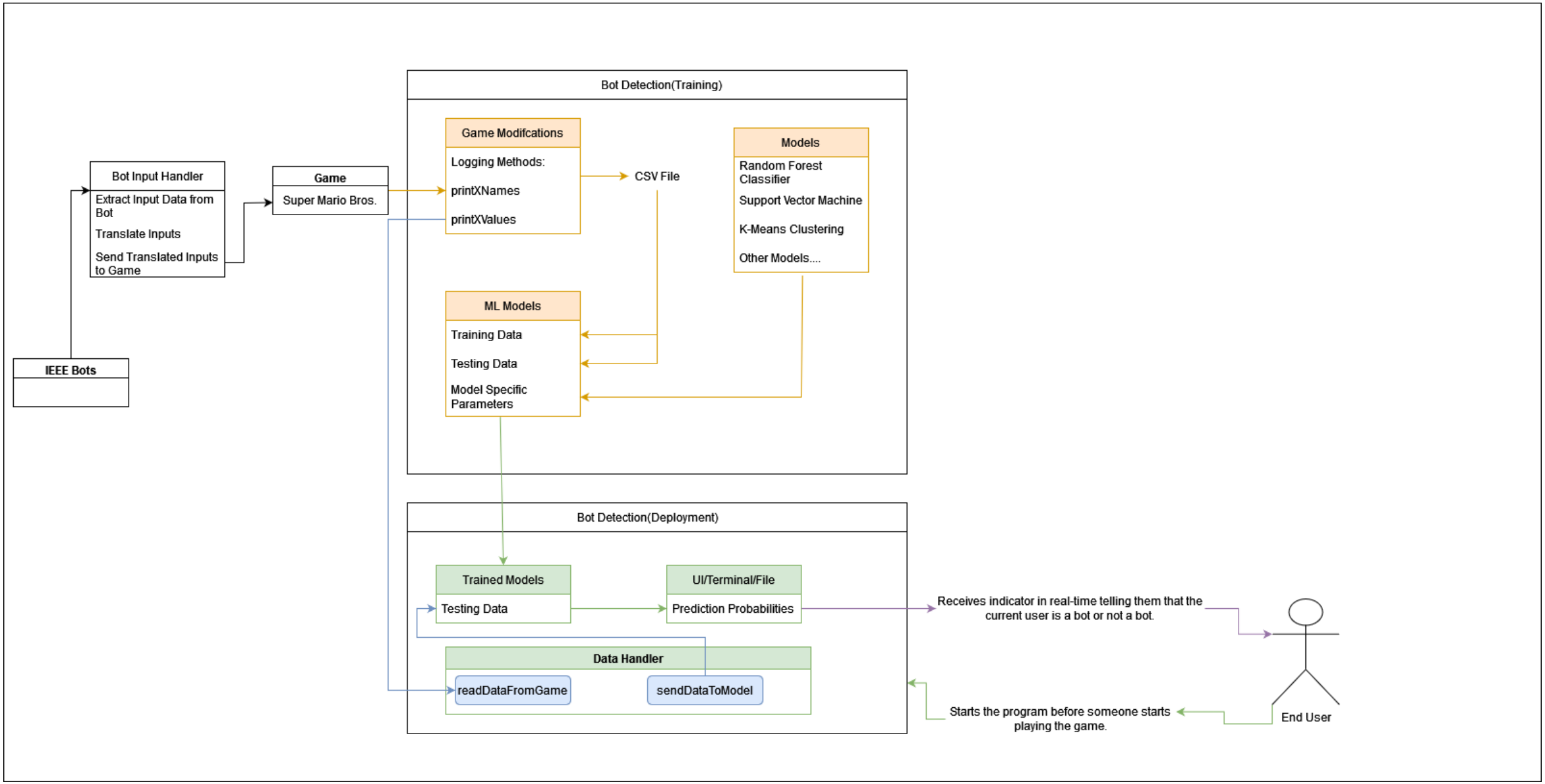
- Data Gathering
  - Stripping memory from running program
- Data Cleaning
  - Clean the data to only keep relevant and effective predictors/features
- Evaluation
  - Statistical Analysis
  - Machine Learning Models
    - Clustering vs. Classification
  - Traditional Algorithms
- Implementation
  - Create live feedback while playing the game



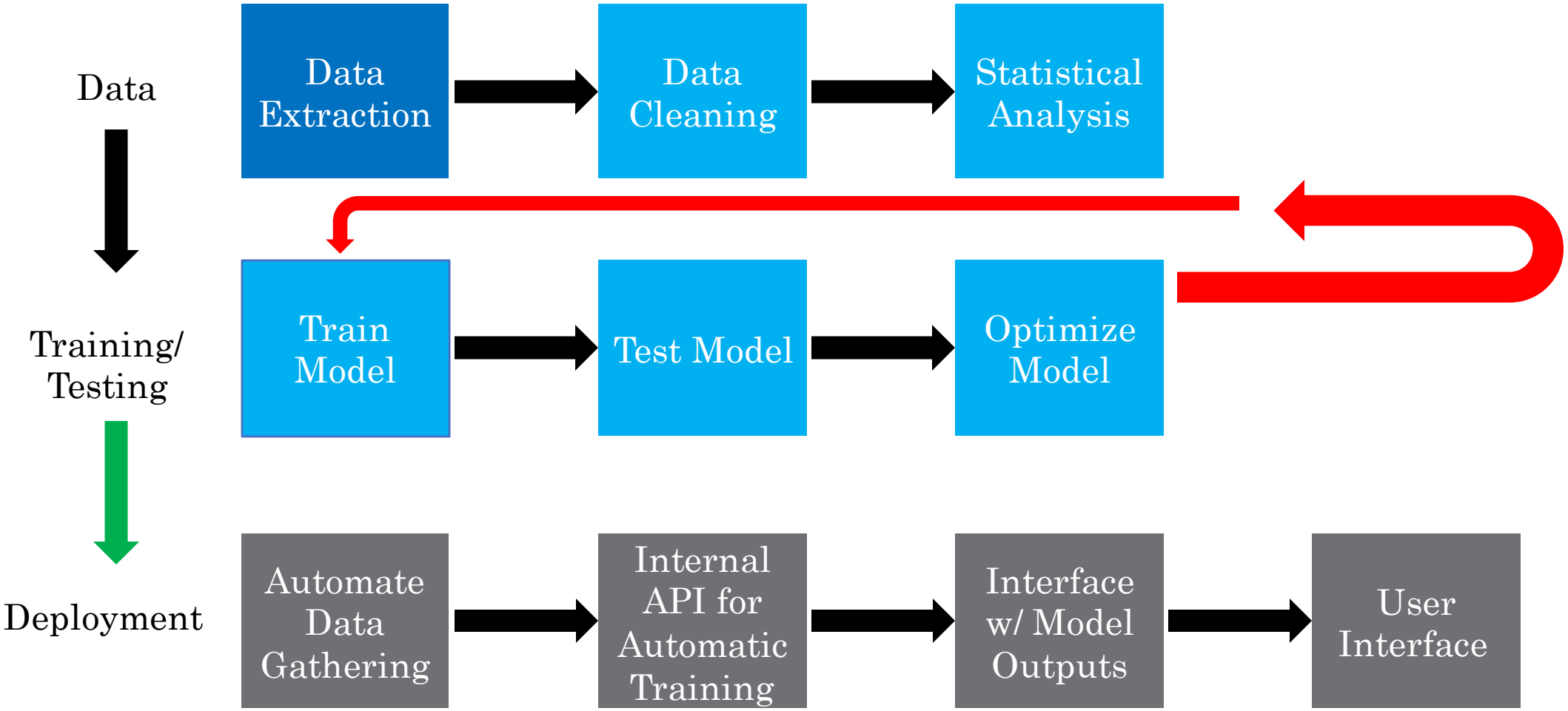
Dependency Graph



Interaction Graph



Implementation Approach



Current Status

- Implemented Memory Scraping in Host Game
  - Bypassed external memory scraping
    - Kernel encrypts memory (on some systems)
    - Logs RAM (2KB) to .csv file
    - Pre-process some labels
      - Game Time (s)
      - Program Time (ms)
      - Controller Input
      - Input Duration
      - X and Y Position
      - ...
- Implemented Data Cleaning
- Implemented Data Analysis
- Implemented Multiple Prediction Functions
  - Comparing two users until bot is implemented

Data Cleaning

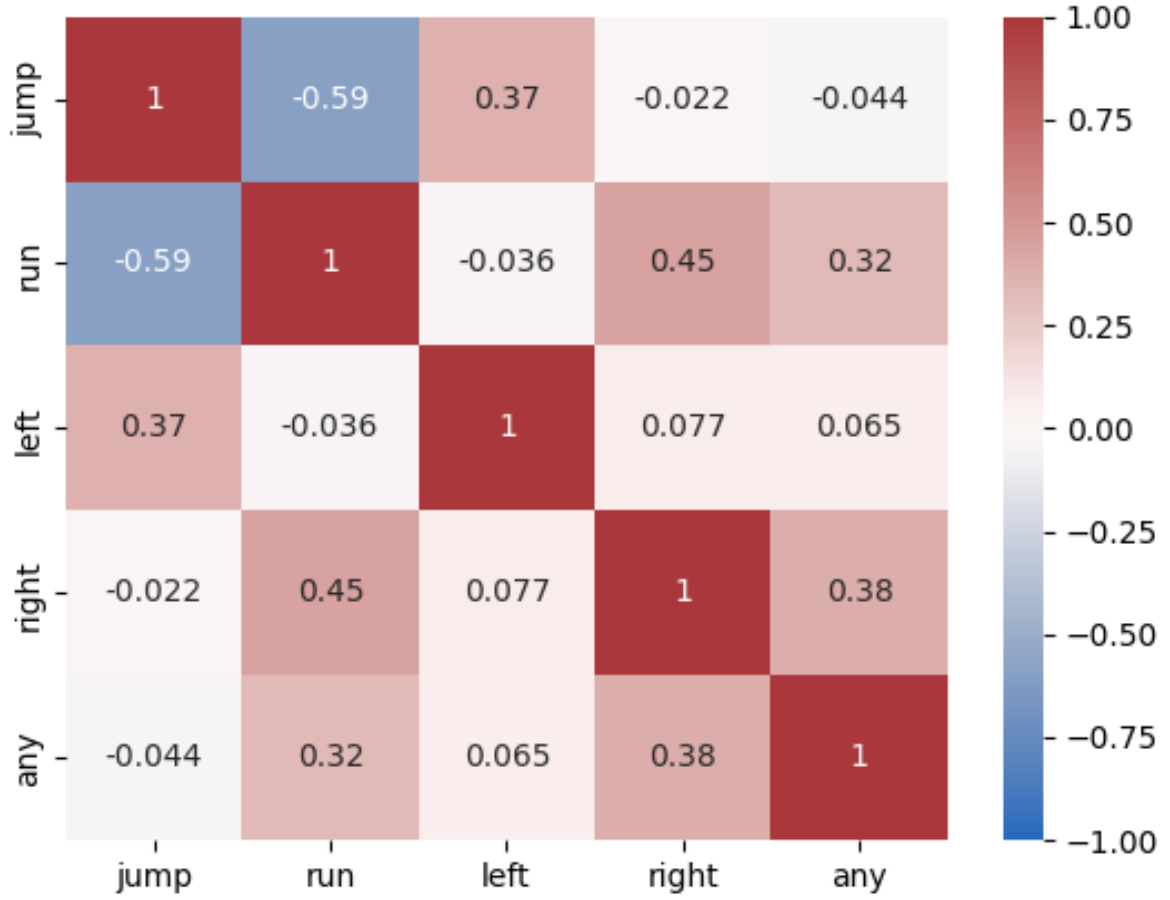
- Remove unnecessary predictors
- Normalize numerical data
  - Different scenarios further normalized data
- Evaluate null values
  - Determine if they should be dropped or modified
- Remove numbers outside standard range

	Name	SDL_Ticks	Timer	PlayerState	CombinedButton	A	B	Select	Start	Up	...	addr_2039	addr_2040	addr_2041	addr_2042	addr_2043	addr_2044	addr_2045	addr_2046	addr_2047	Unnamed: 2071
0	3	74	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	165.0	NaN
1	3	91	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	165.0	NaN
2	3	107	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	165.0	NaN
3	3	123	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	165.0	NaN
4	3	140	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	165.0	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2955	3	49309	219	5	0	0	0	0	0	0	...	0	2	1	9	0	0	0	0	165.0	NaN
2956	3	49325	218	5	0	0	0	0	0	0	...	0	2	1	8	0	0	0	0	165.0	NaN
2957	3	49341	217	5	0	0	0	0	0	0	...	0	2	1	7	0	0	0	0	165.0	NaN
2958	3	49357	216	5	0	0	0	0	0	0	...	0	2	1	6	0	0	0	0	165.0	NaN
2959	3	49376	215	5	0	0	0	0	0	0	...	0	2	1	5	0	0	0	0	165.0	NaN

```
df.isna().sum()
✓ 0.0s
Name                0
SDL_Ticks           0
Timer               0
PlayerState         0
CombinedButton      0
A                   0
B                   0
Select              0
Start               0
Up                  0
Down                0
Left                0
Right               0
RightDuration       0
LeftDuration        0
JumpDuration        0
RunDuration         0
TotalJumpAirtime    87
TotalFallAirtime    0
POWER_EXIST         0
FACE                0
X                   0
Y                   0
dtype: int64
```

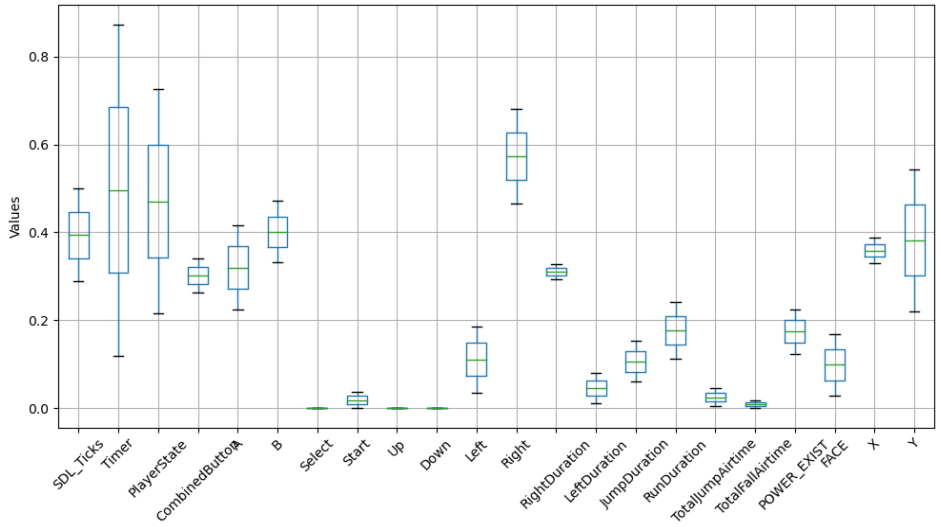
Correlation Analysis

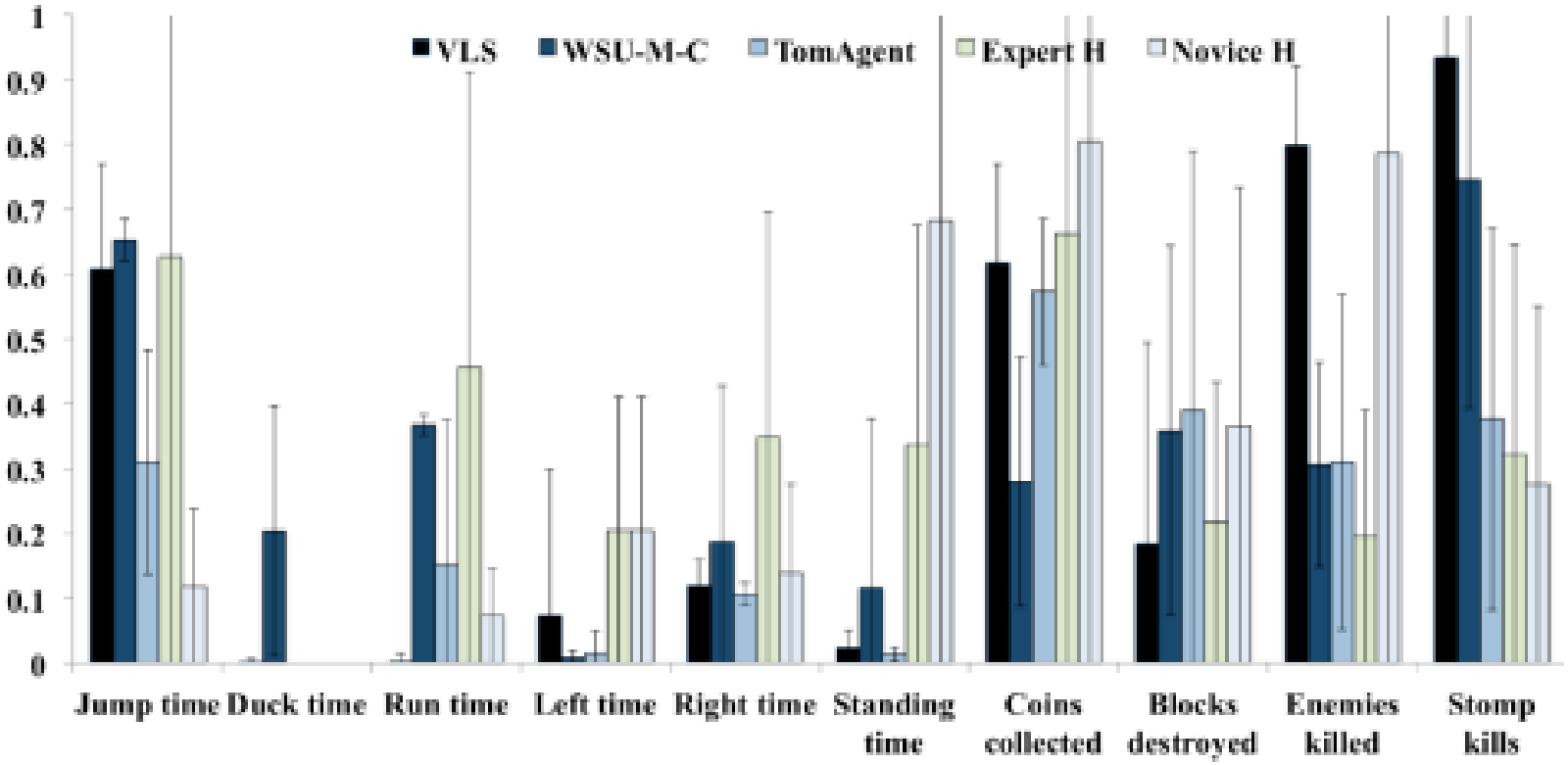
- Taking averages from various statistics such as how long between inputs.
- Using these averages to try and find a correlation between inputs
- Different correlation methods:
  - Pearson (linear)
  - Kendall
  - Spearman
- Best results, Spearman (depicted in graph)



Player Analysis

- Modeled results of peer-reviewed paper
- Applied max-min normalization
- Found average and standard deviation for each labeled predictor

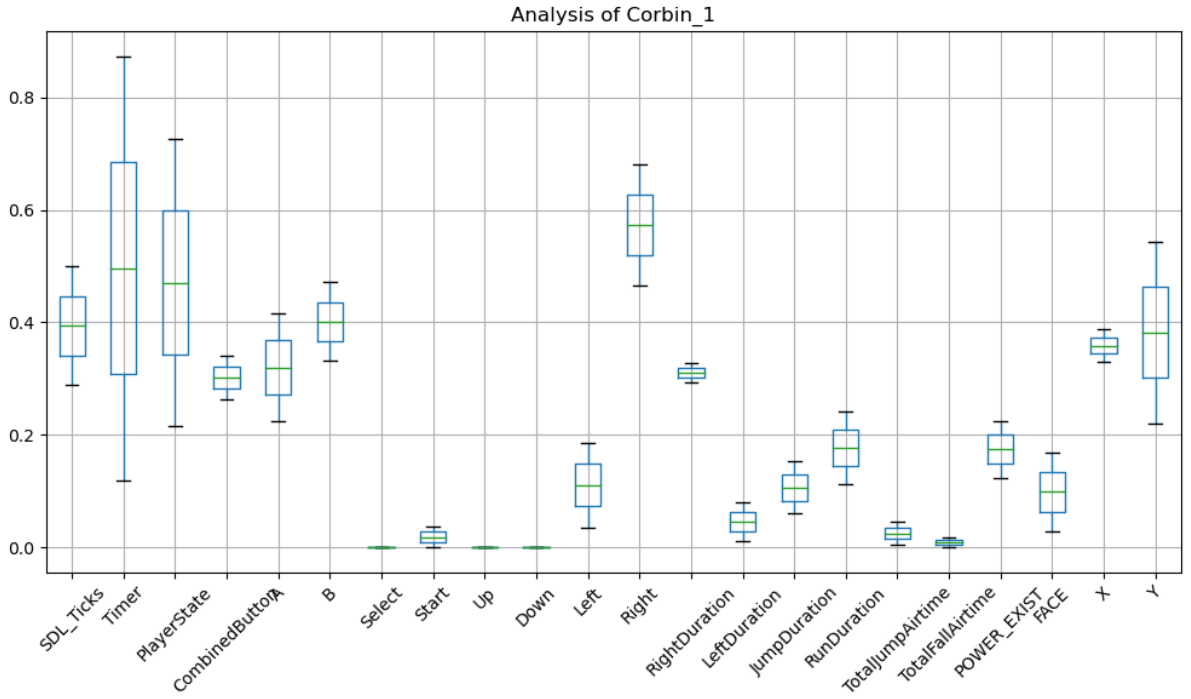


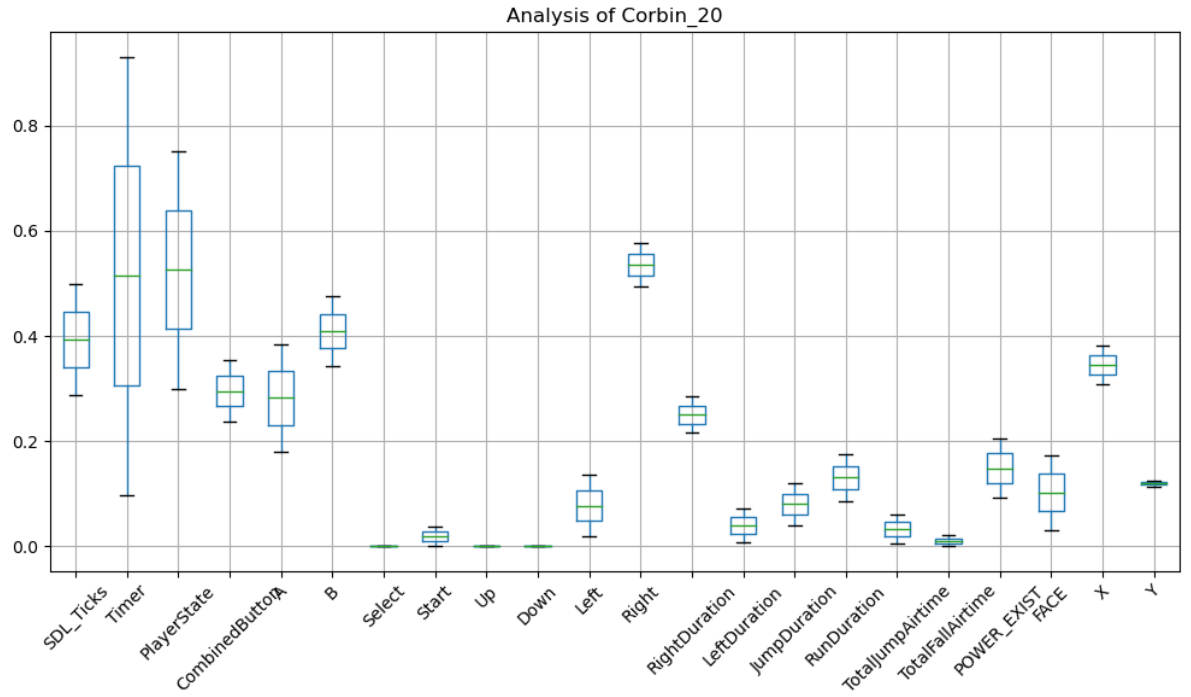


N. Shaker *et al*., "The turing test track of the 2012 Mario AI Championship: Entries and evaluation," *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, Niagara Falls, ON, Canada, 2013, pp. 1-8, doi: 10.1109/CIG.2013.6633634.

Improvement Trajectory

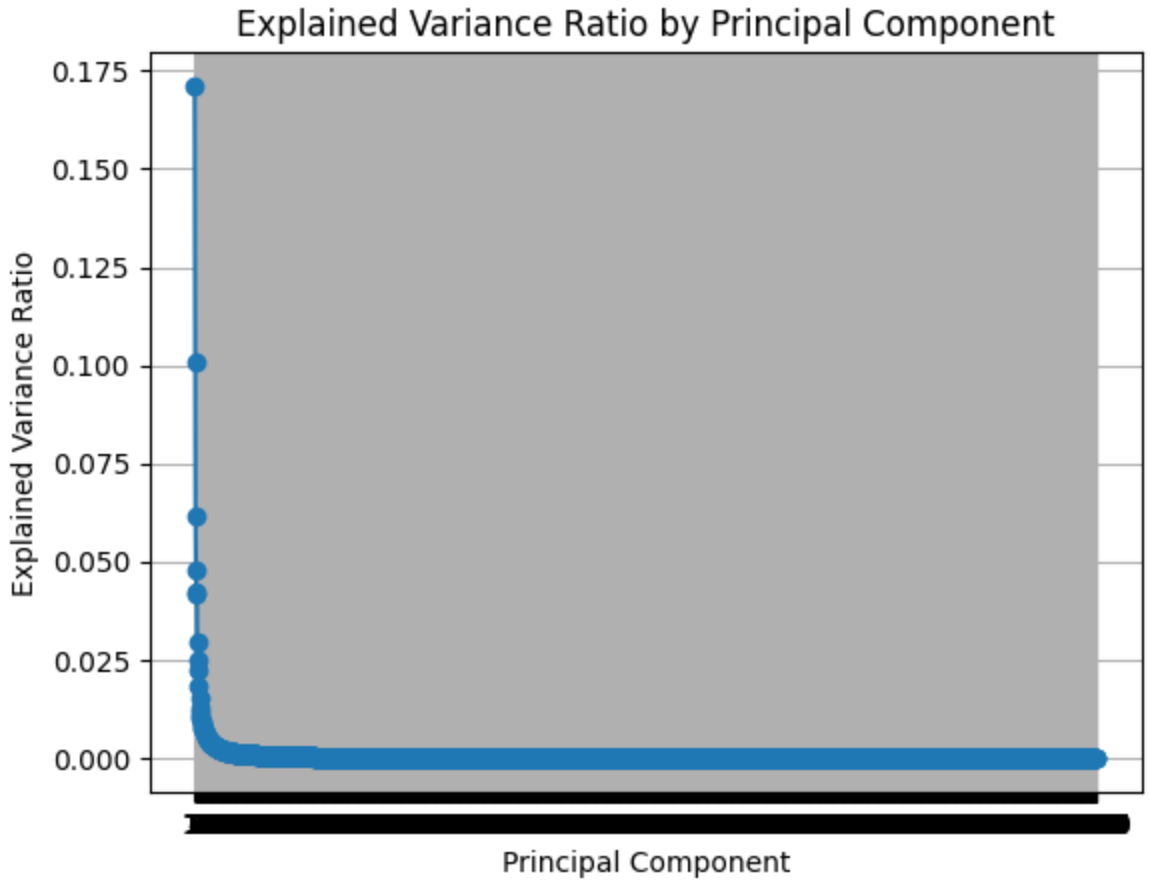
- How does the average player improve as they play more?
- Does a non-genetic bot improve in the same way?
- Can we compare the first run and the last run to determine an improvement level?





Principal Component Analysis (PCA)

- Find the 'elbow'
- Visual analysis indicates our best ratio
- We then reduce the number of dimensions based on this ideal



Models

Logistic Regression

Random Forest Classifier

Support Vector Machine

K-Means Clustering

KNN Clustering

LSTM

Hidden Markov Models

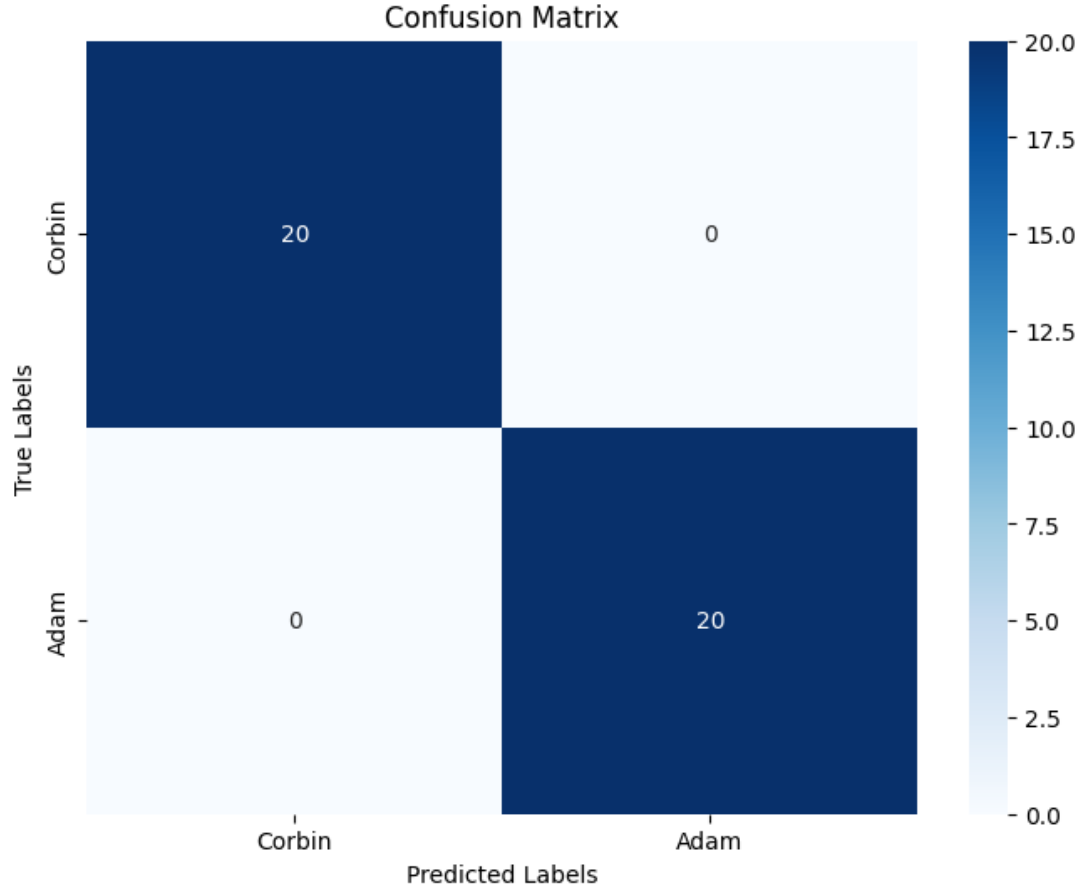
KNN Clustering / Classification

- Challenges
  - Large range of accuracy when predicting against different users
  - Difficult to adapt
- K=2
  - Binary classification comparing two users
- K=5
  - Cluster All Users

K=2	Bot	Adam	Carlos	Corbin	Jose	Nhan
Bot	-					
Adam		-	100%	99%	48%	52%
Carlos		100%	-	100%	100%	100%
Corbin		99%	100%	-	46%	51%
Jose		48%	100%	46%	-	48%
Nhan		52%	100%	51%	48%	-

Hidden Markov Model

- Each user gets their own HMM which will be adjusted as they play the game to model their improvement trajectory
- Transitions are given a baseline then adjusted from the baseline
- Output: ('1', 82.9877539623103) for the Adam model trained on first run and tested on second run



Logistic Regression

Rationale:

- Very simple
- A starting point after collecting data

Challenges:

- Only works for binary classification.
- Only works on data with linear relationships.

Logistic Regression w/ Scoring Function

Samples	Prediction	Confidence
Carlos_41	Jose	55.9%
Carlos_42	Jose	57.0%
Carlos_43	Jose	57.2%
Carlos_44	Jose	58.0%
Carlos_45	Jose	60.0%

Samples	Prediction	Confidence
Jose_41	Jose	74.5%
Jose_42	Jose	74.2%
Jose_43	Jose	74.8%
Jose_44	Jose	75.3%
Jose_45	Jose	74.5%

Support Vector Machines

Rationale:

- More robust than logistic regression
- Can choose between linear, polynomial, rbf, or sigmoid kernel.

Challenges:

- Incredibly slow training and testing time.
- For this dataset, does not provide good enough accuracy.
- Had to create a scoring function to reduce dimensions.

SVM (Polynomial, Degree = 3) w/ Scoring Function

Samples	Prediction	Confidence
Carlos_41	Carlos	75.4%
Carlos_42	Carlos	77.9%
Carlos_43	Carlos	79.1%
Carlos_44	Carlos	79.6%
Carlos_45	Carlos	76.0%

Samples	Prediction	Confidence
Jose_41	Jose	57.8%
Jose_42	Jose	55.5%
Jose_43	Jose	53.4%
Jose_44	Jose	55.9%
Jose_45	Jose	53.0%

Random Forest Classifier

Rationale:

- Recommended by client
- Non-Parametric
- Helps identify feature importance
- Faster training time than SVM for the size of the dataset

Random Forest Classifier (Binary)

Person 1 Sample

Frame	Prediction
14842	[0.65835615 0.34164385]
14858	[0.710278 0.289722]
14875	[0.96485905 0.03514095]
…	

Person 2 Sample

Frame	Prediction
14842	[0.29875938 0.70124062]
14858	[0.30117425 0.69882575]
14875	[0.43758544 0.56241456]
…	

Random Forest Classifier (Binary)

Samples	Confidence
Carlos_41	99.2%
Carlos_42	100.0%
Carlos_43	99.9%
Carlos_44	100.0%
Carlos_45	99.3%
Carlos_46	99.9%
Carlos_47	99.9%
Carlos_48	99.8%
Carlos_49	98.3%
Carlos_50	99.9%

Samples	Confidence
Adam_16	80.8%
Adam_17	88.0%
Adam_18	86.0%
Adam_19	75.1%
Adam_20	88.9%

Random Forest Classifier (Multi-Class)

Challenges:

- Expanding RFC to multi-class makes it much less accurate.

Random Forest Classifier (Multi-Class)

Samples	Prediction	Confidence
Carlos_41	Carlos	65.4%
Carlos_42	Carlos	66.0%
Carlos_43	Carlos	54.2%
Carlos_44	Carlos	68.8%
Carlos_45	Carlos	65.2%

Samples	Prediction	Confidence
Adam_16	Nhan	61.2%
Adam_17	Nhan	69.0%
Adam_18	Nhan	63.1%
Adam_19	Nhan	51.5%
Adam_20	Nhan	65.7%

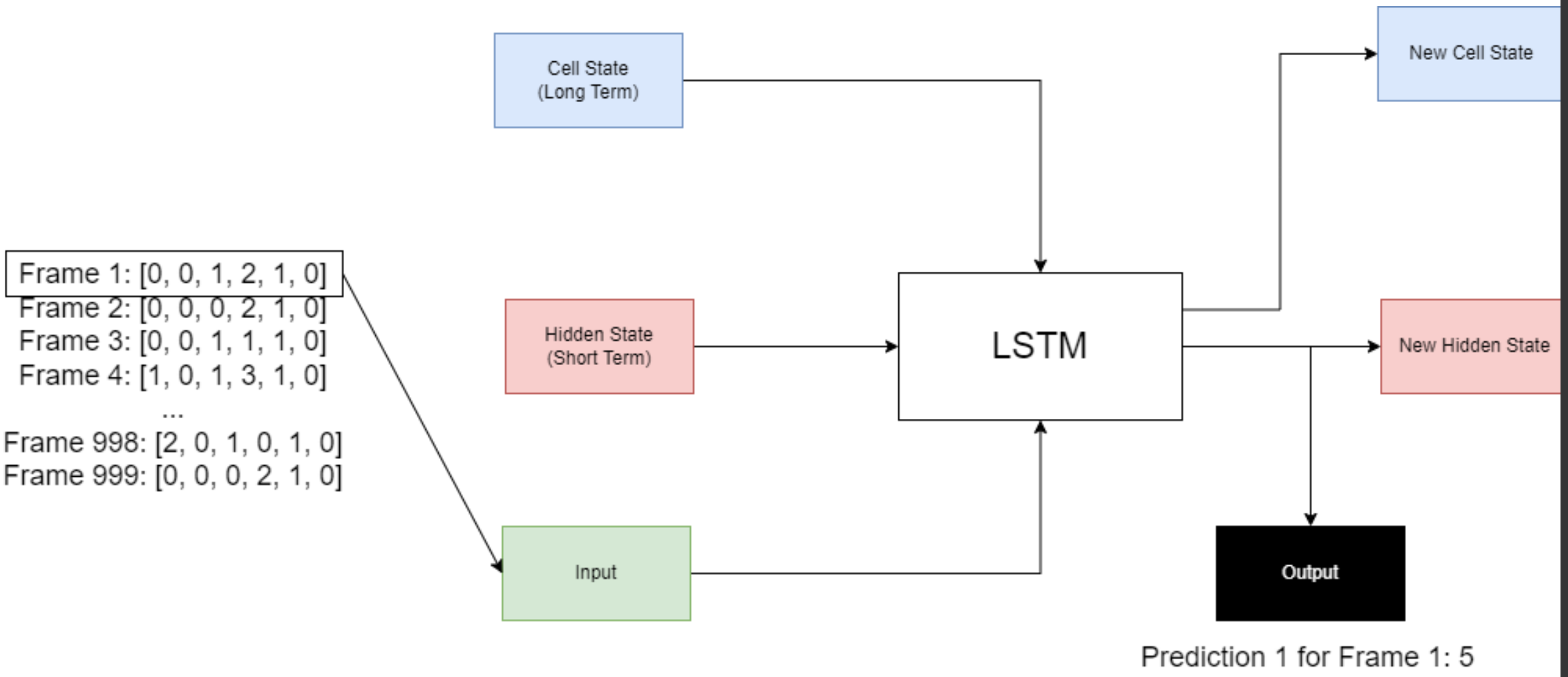
Long-Short Term Memory

RNN with additional cell states for long term memory

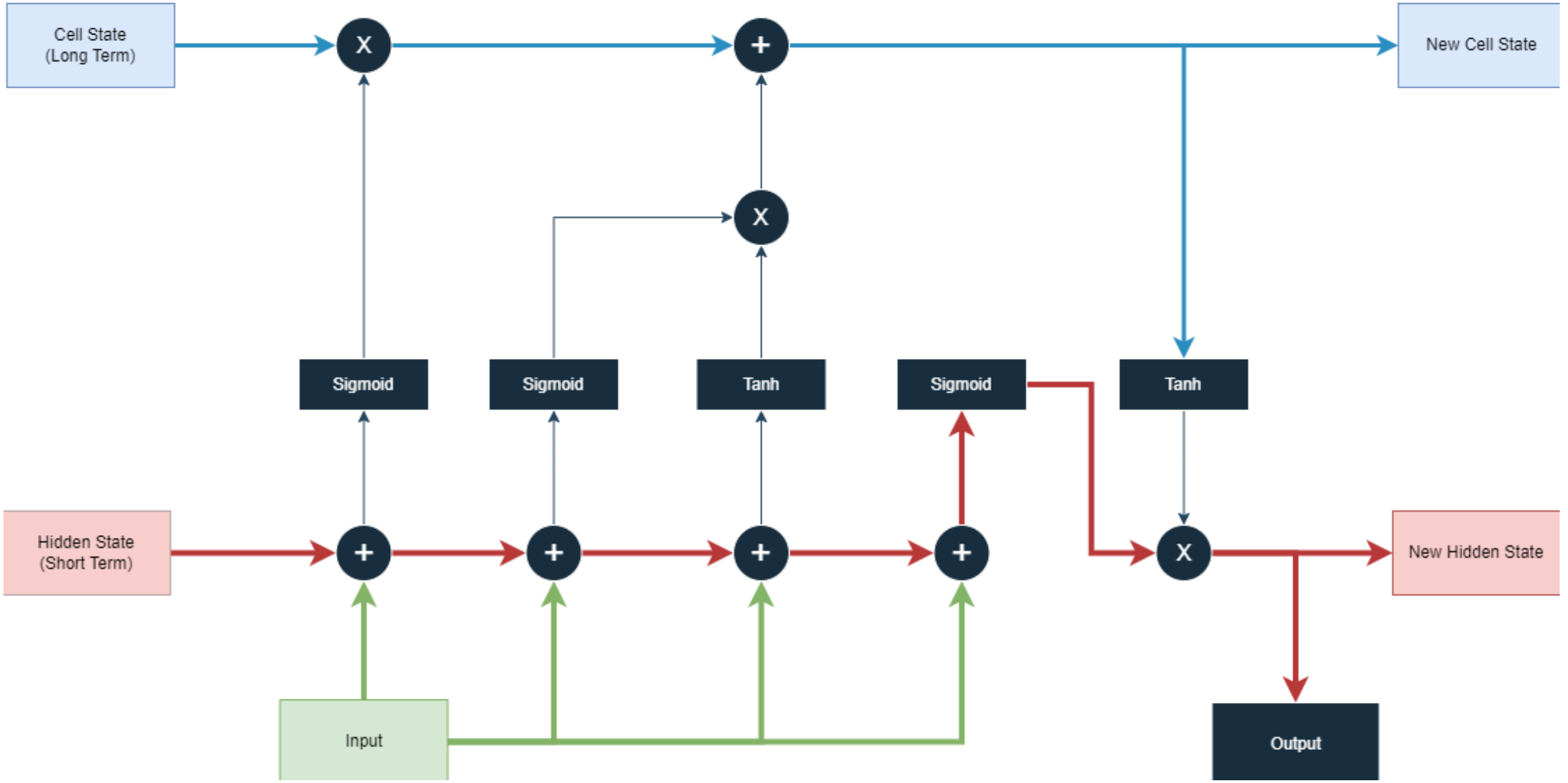
Training on sequence of data one by one

Rationale: Gameplay is a sequence of data

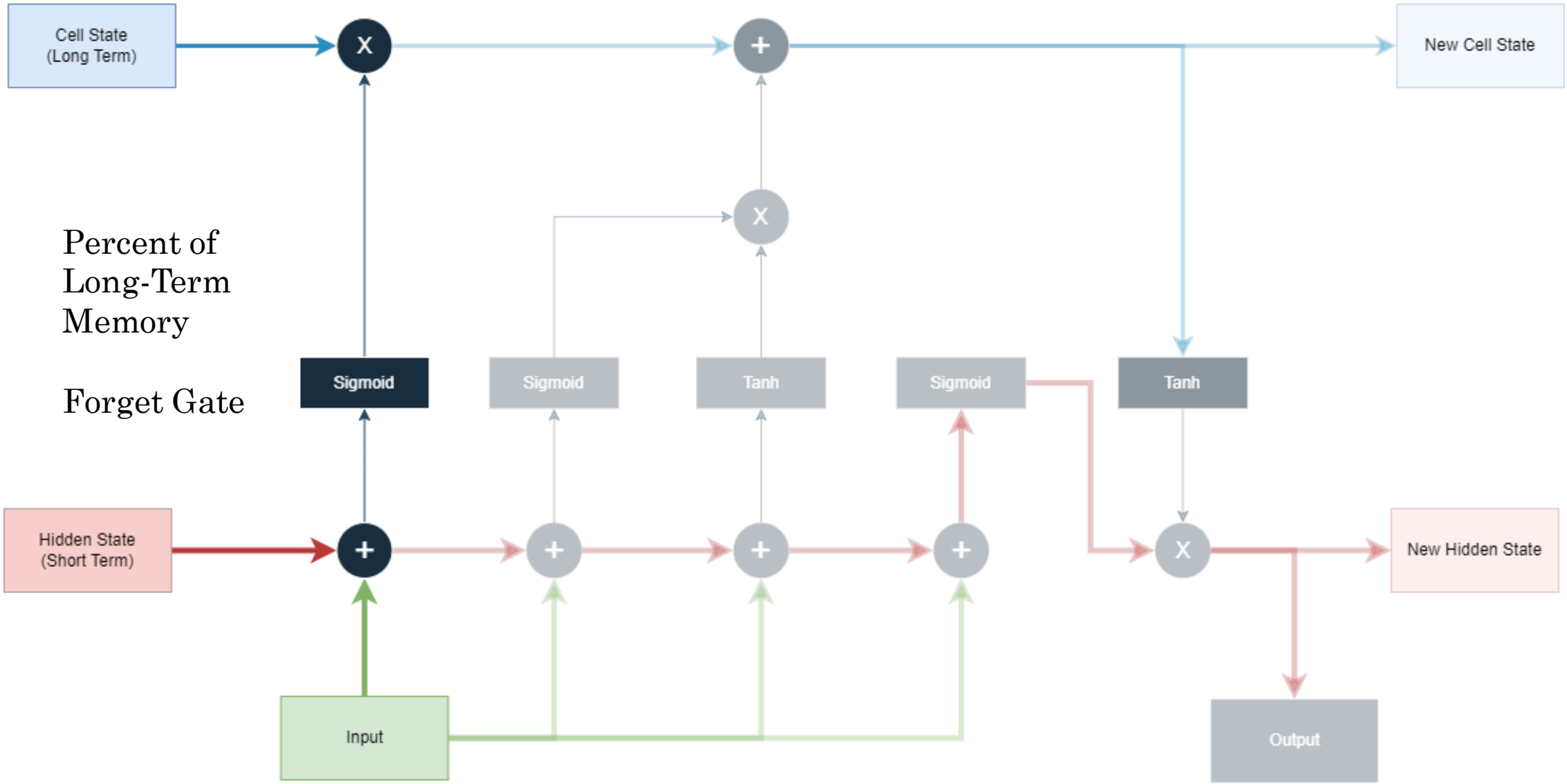
LSTM Architecture



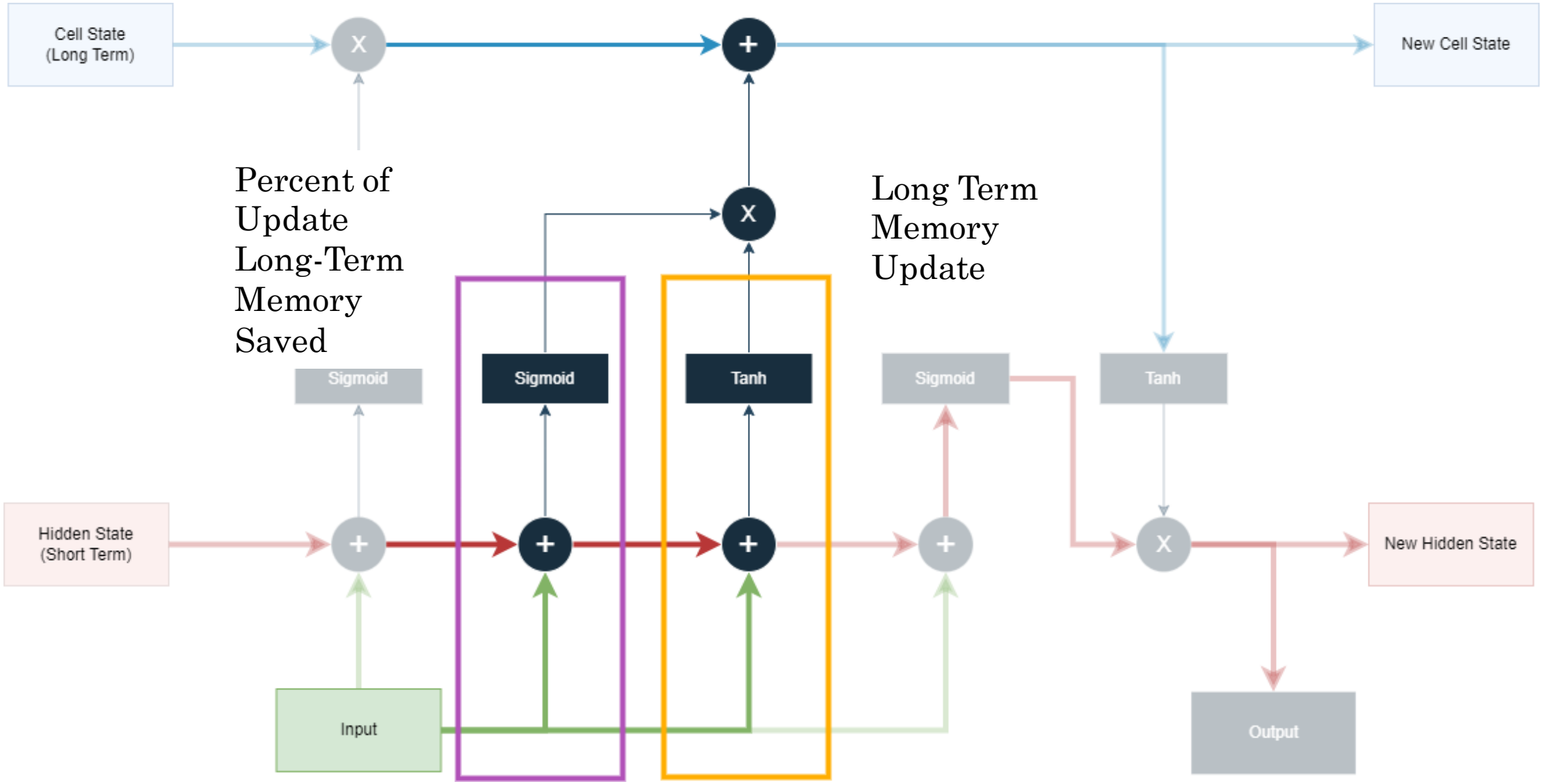
LSTM Architecture



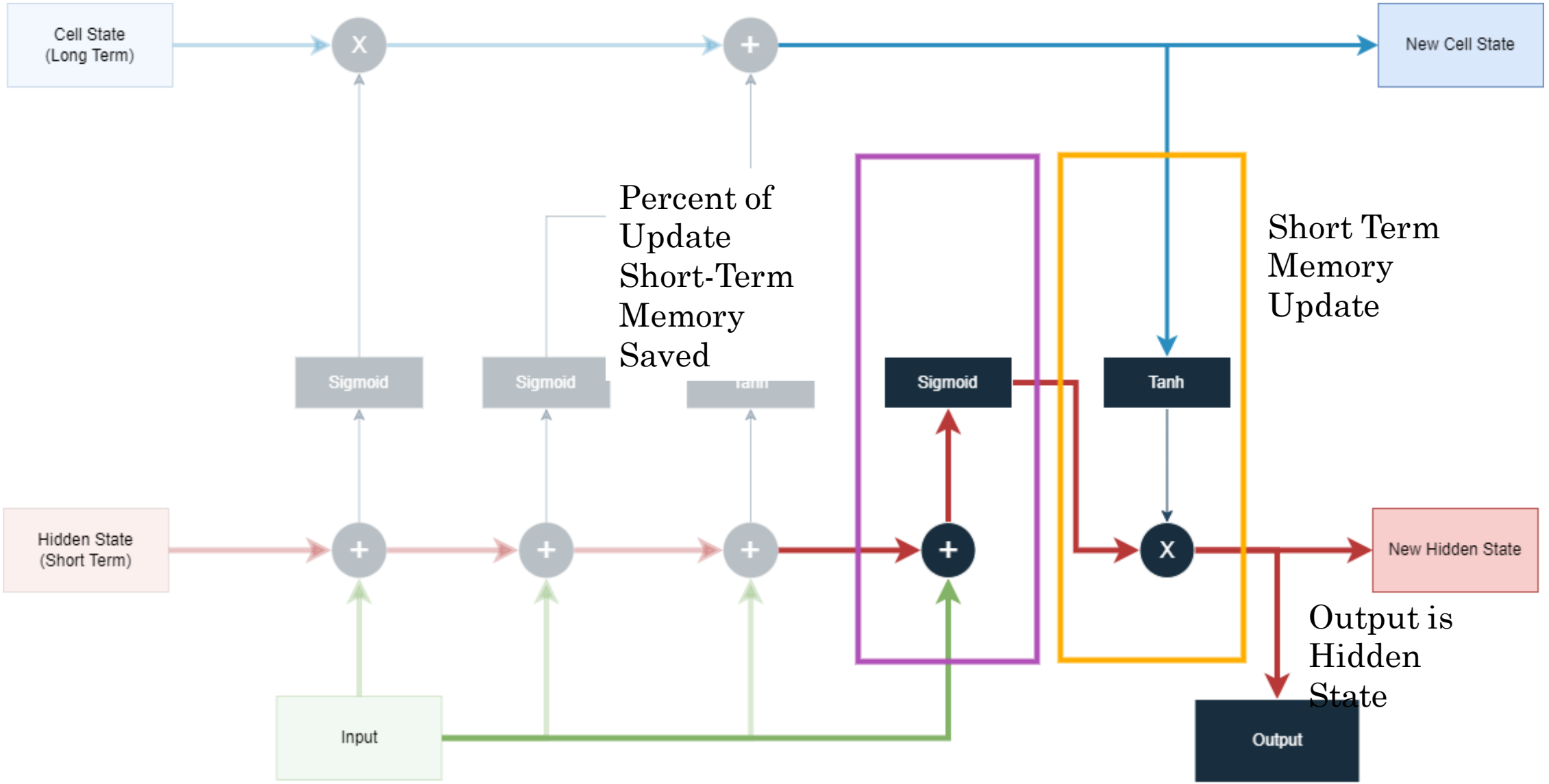
LSTM Architecture



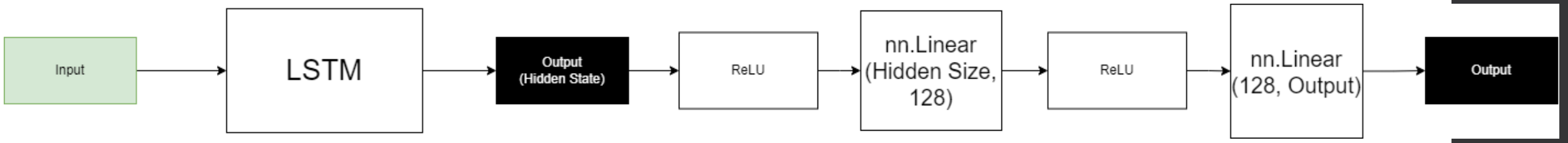
LSTM Architecture



LSTM Architecture

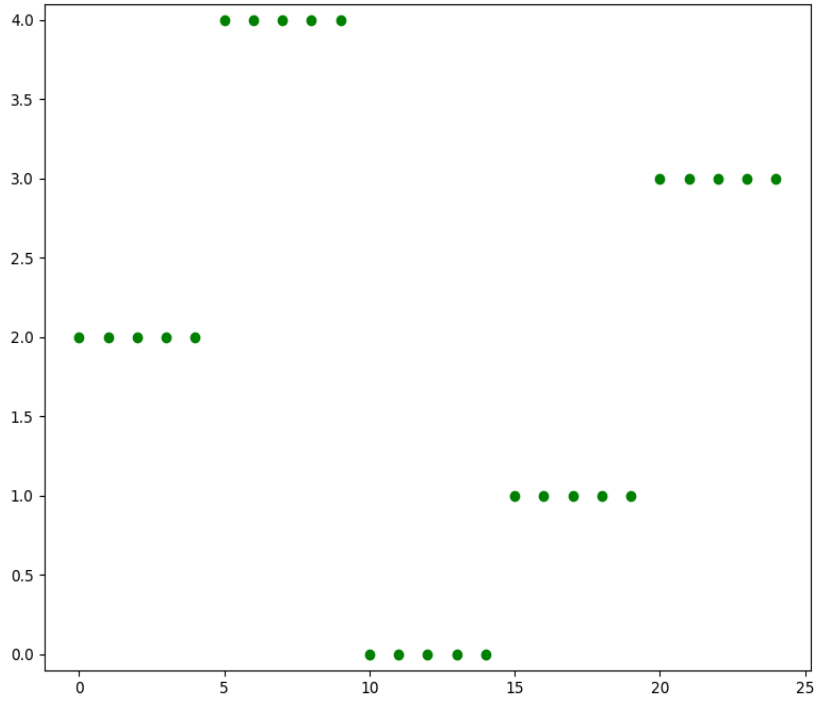


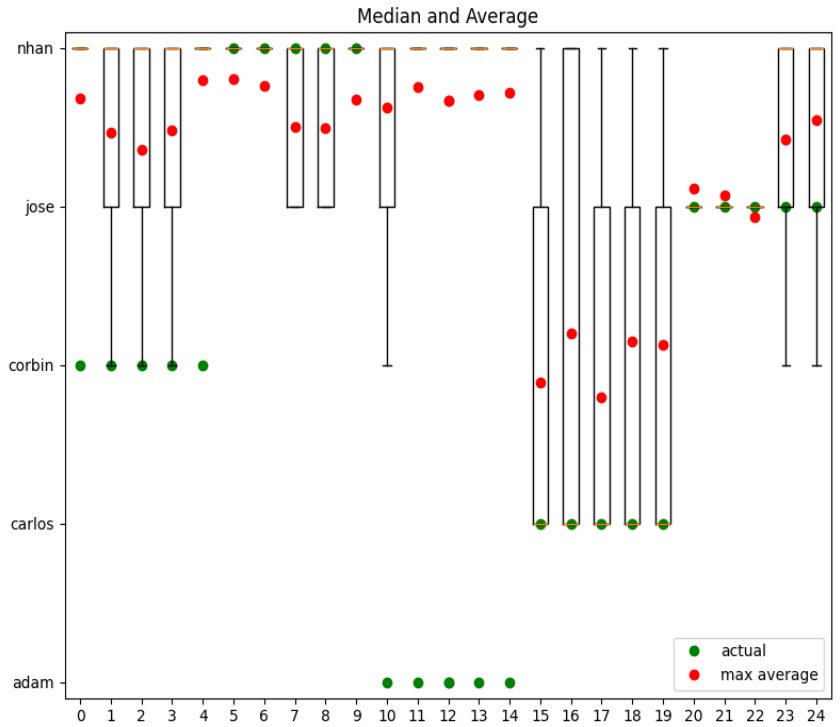
LSTM Architecture

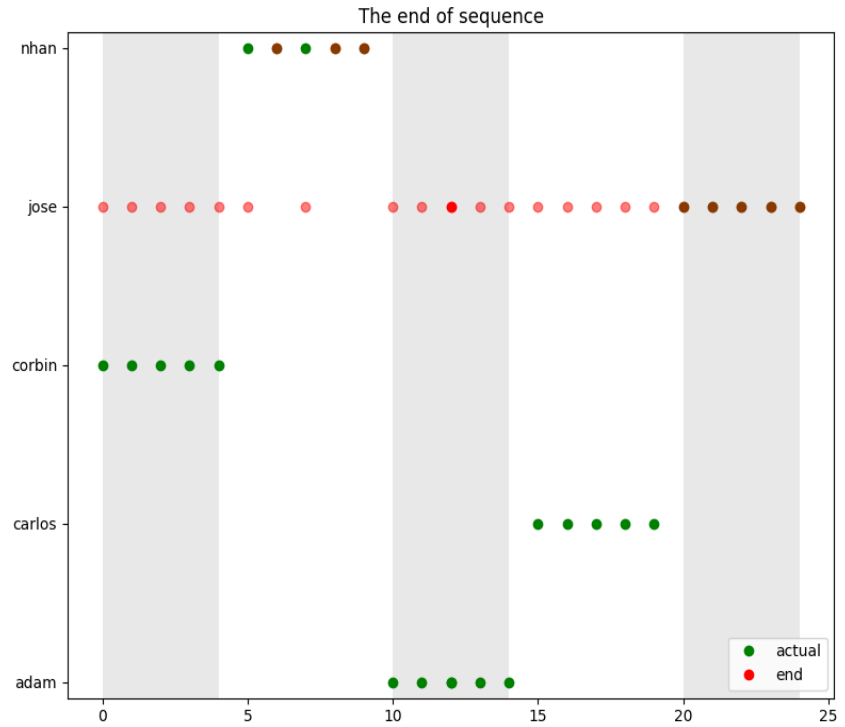


Long-Short Term Memory Full Sequences

FEATURES: X, Y, RIGHT, LEFT, A, B, PLAYER_STATE, FACE

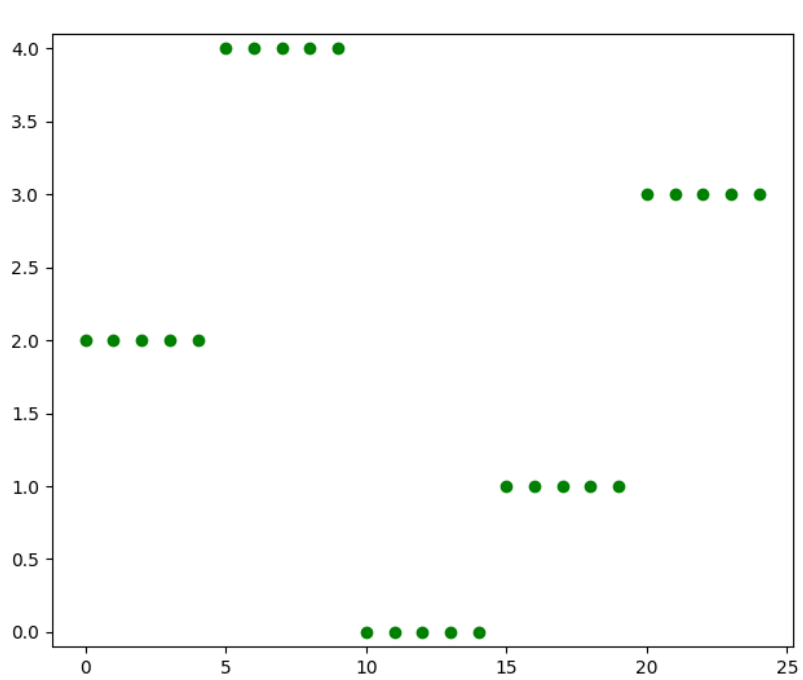


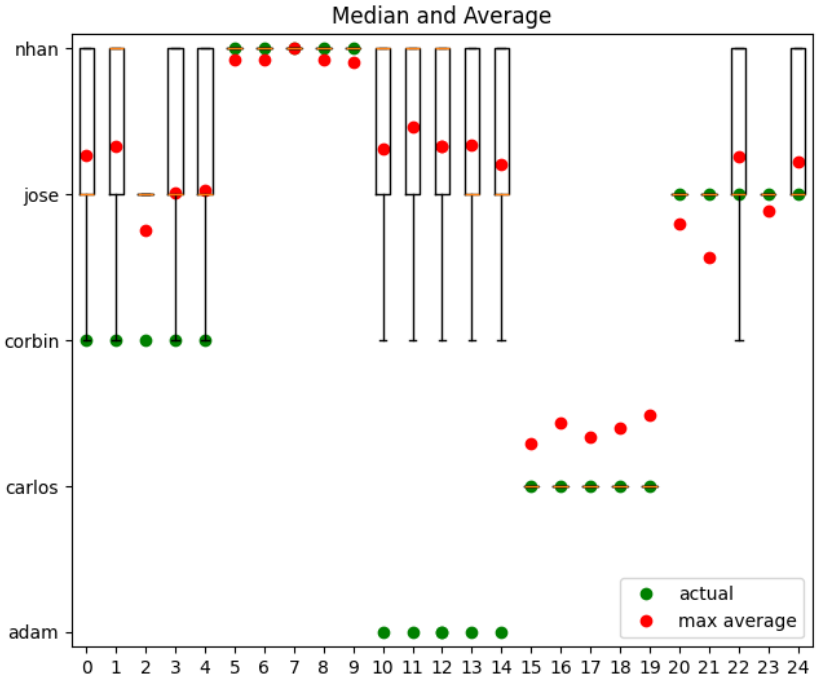


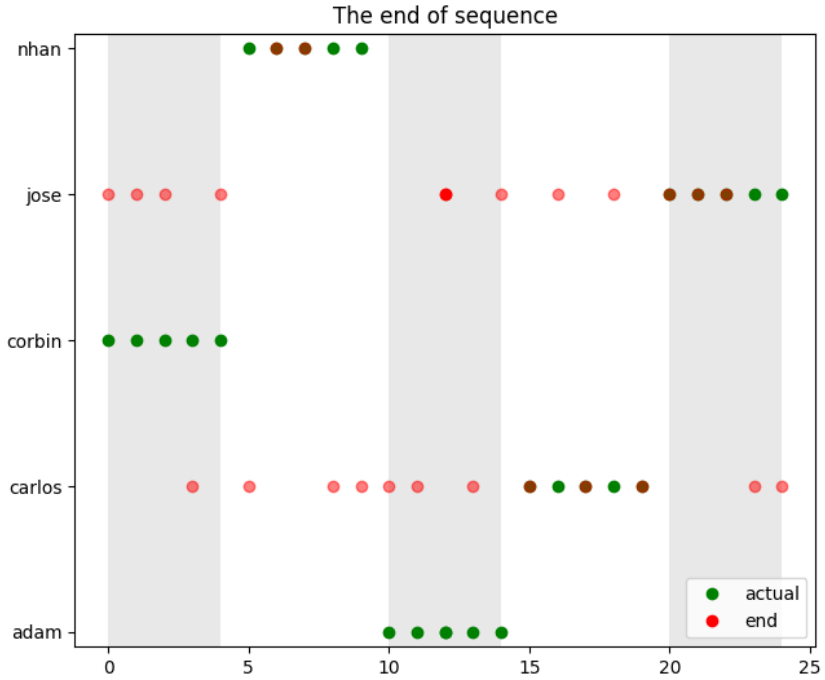


Long-Short Term Memory Event Sequences

FEATURES: DELTA_TICKS, X, Y, RIGHT, LEFT, A, B, PLAYER_STATE, FACE

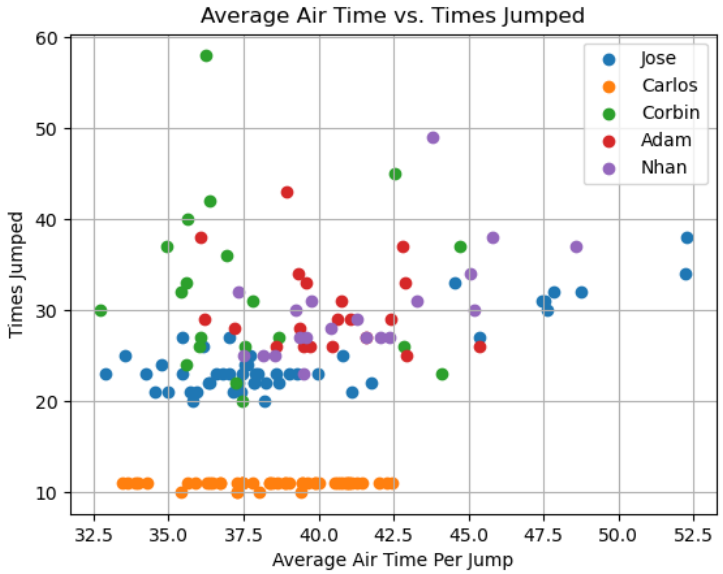


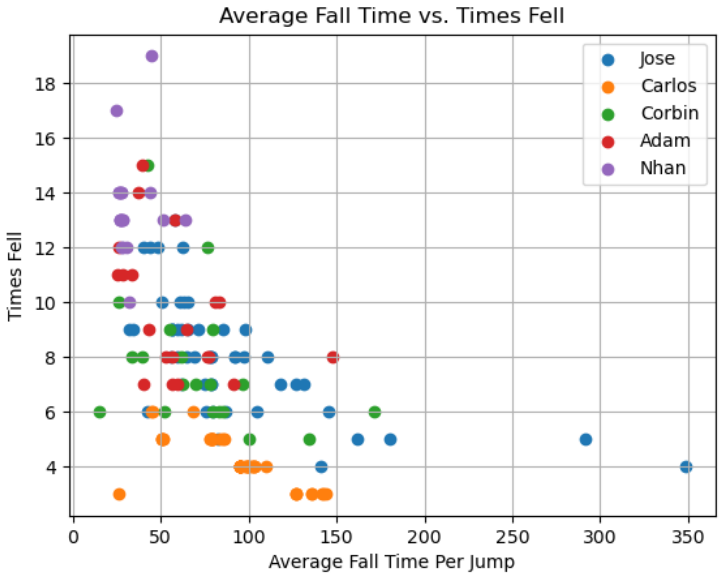


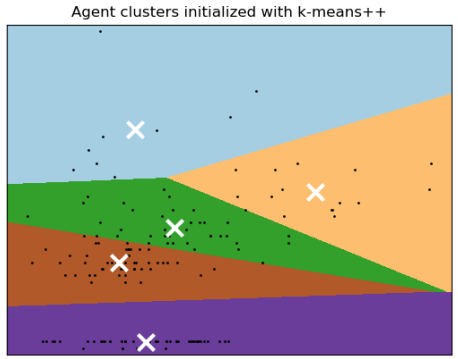


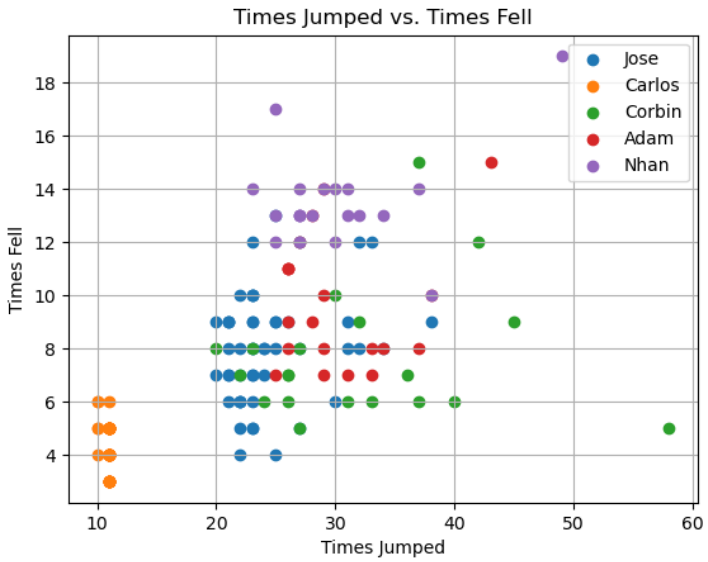
K-Means Clustering

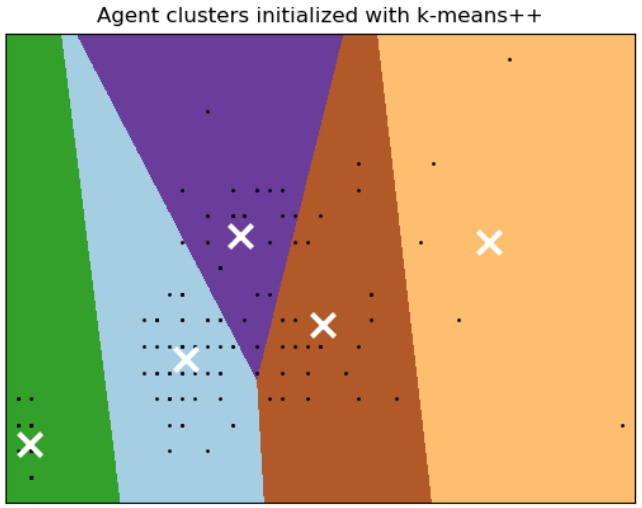
- Limited use unless data is preprocessed and/or feature selected
- Performance measured by similarity to labels (ground truth)



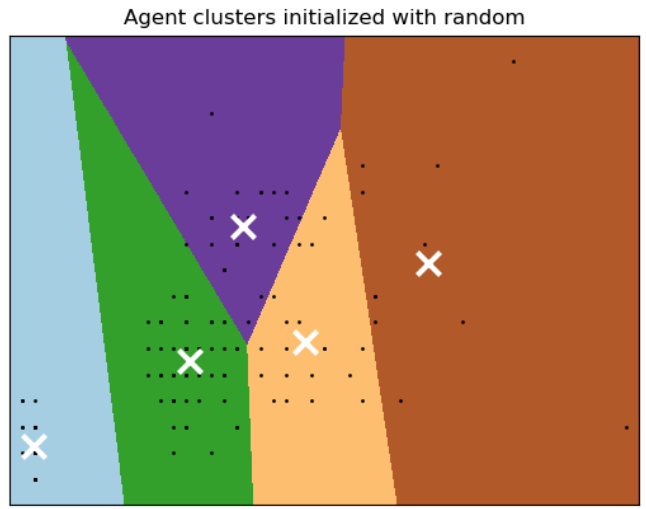




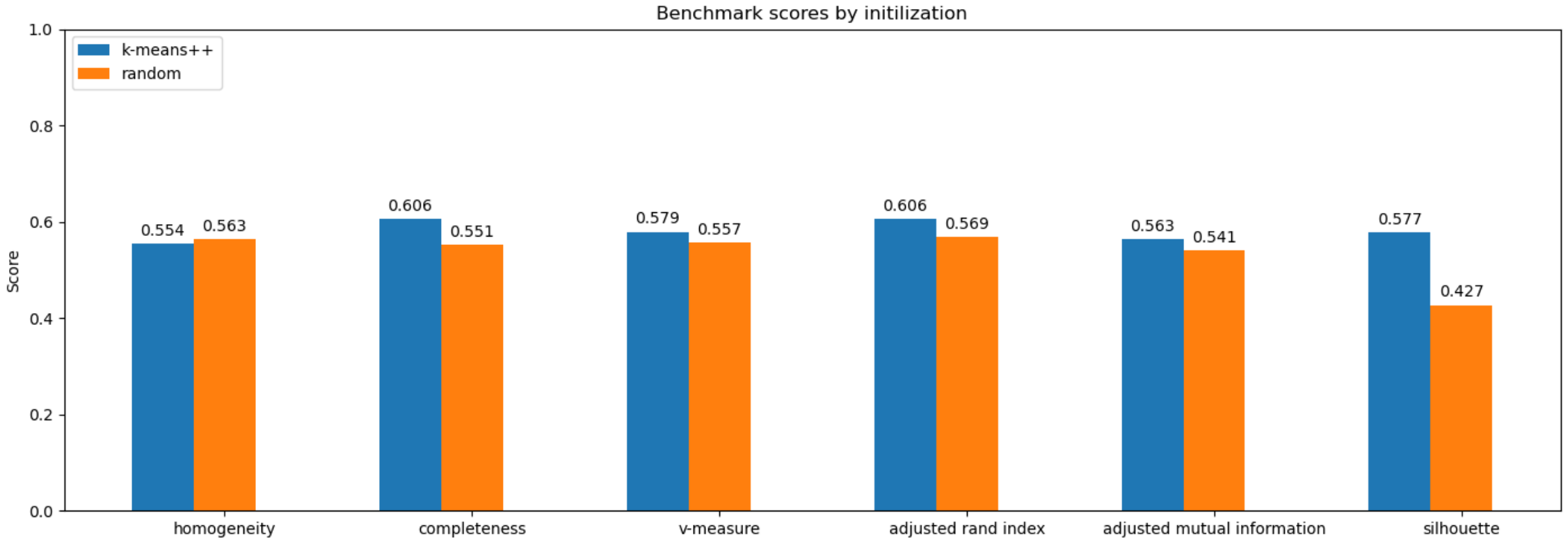




Inertia: 41



Inertia: 42



Challenges with K-Means

- Arbitrary selection of features
- Information loss
- More agents?
- Different game?

Challenges with Data

- Had to produce our own data
- Data cleaning/preprocessing needed
- Process of trial and error

Challenges with Models

- Which model will meet which requirements?
- Model selection (which will be most effective for our data?)
- How to choose proper parameters for models?

Remaining Work and Timeline

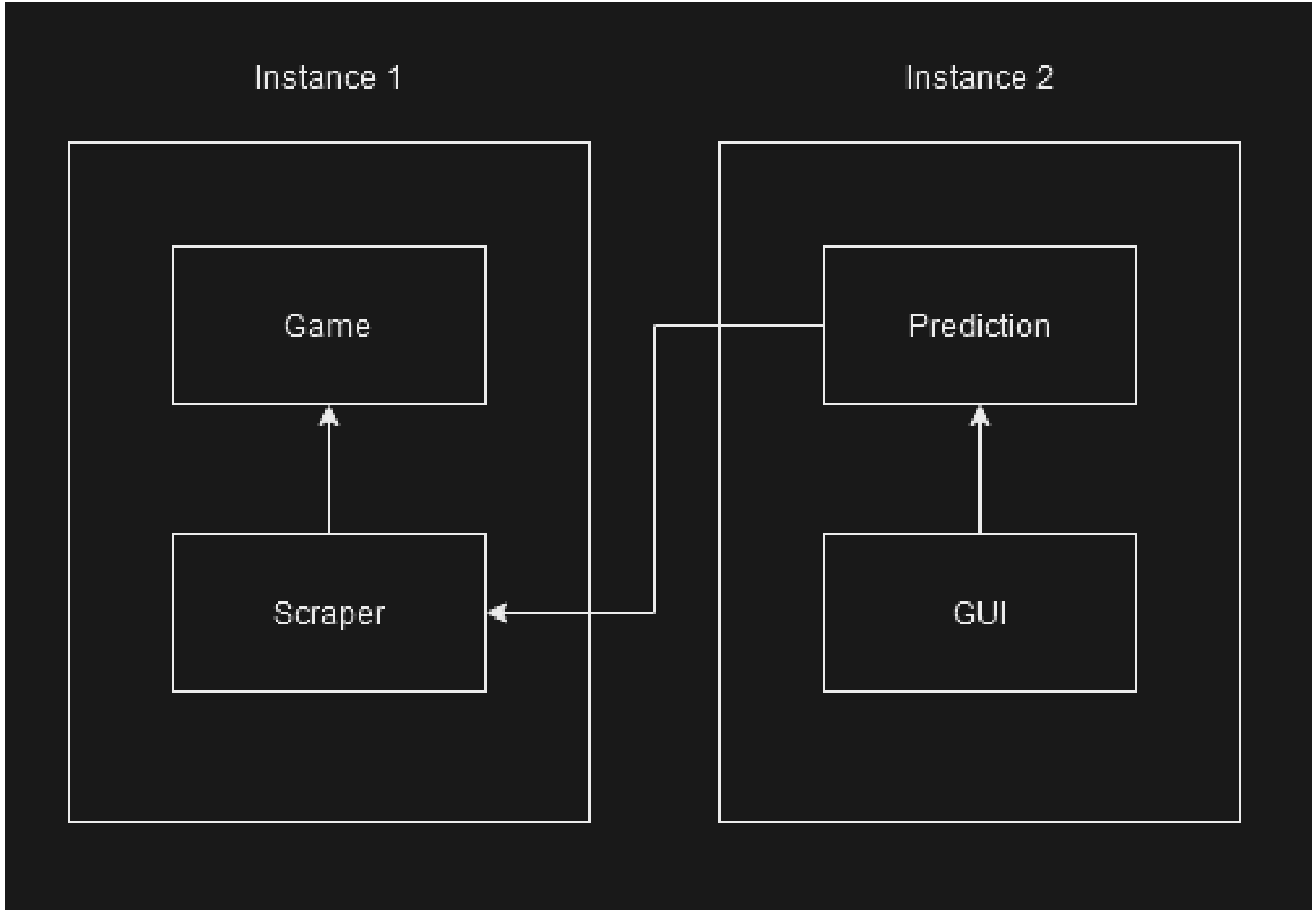
- Collecting bot data from IEEE Mario Turing Test Competition Entries
- Adjusting existing models to interact with the new data
- Creating new models to address several bot play styles
- Generalize findings
- Client-side bot detection in real time

Demo

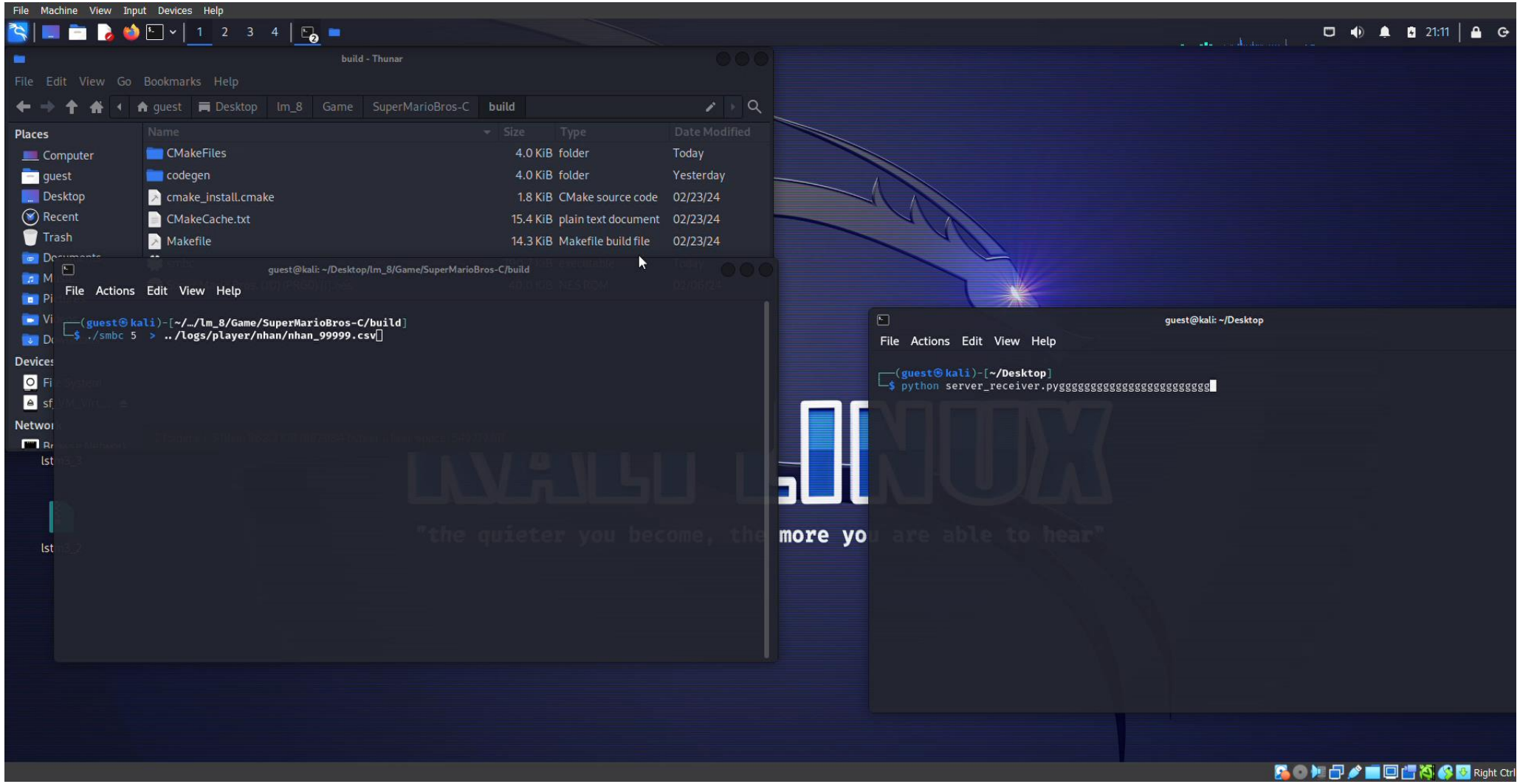
Current Data to Prediction Pipeline

- Game Data to Prediction
- Socket IO
  - Game: C++
  - Prediction: Python

Current Data to Prediction Pipeline



Demo Video


File Machine View Input Devices Help
build - Thunar
File Edit View Go Bookmarks Help
guest Desktop lm_8 Game SuperMarioBros-C build
Places
Computer
guest
Desktop
Recent
Trash
Name Size Type Date Modified
CMakeFiles 4.0 KiB folder Today
codegen 4.0 KiB folder Yesterday
cmake_install.cmake 1.8 KiB CMake source code 02/23/24
CMakeCache.txt 15.4 KiB plain text document 02/23/24
Makefile 14.3 KiB Makefile build file 02/23/24
guest@kali: ~/Desktop/lm_8/Game/SuperMarioBros-C/build
File Actions Edit View Help
(guest㉿kali)-[~/…/lm_8/Game/SuperMarioBros-C/build]
$ ./smbc 5 > ../logs/player/nhan/nhan_99999.csv
guest@kali: ~/Desktop
File Actions Edit View Help
(guest㉿kali)-[~/Desktop]
$ python server_receiver.pyggggggggggggggggggggggg
21:11
Right Ctrl