NGUARD： A Game Bot Detection Framework for NetEase MMORPGs

KDD 2018

目录 OUTLINE

01 Introduction

02 Dataset description

03 The framework

04 Experiments

01

MMORPG

online games in which thousands of players use characters with specific roles to interact with each other and performs adventure-related tasks in the same continuous and persistent world.

01

信息标题/物品类型/游戏/区/服	价格排序	库存
保 全服唯一71固防护腕【在线发货，方便快捷】 有图 信用等级： 物品种类：装备/护腕 游戏/区/服：逆冰塞OL/将进酒/武林天骄	666.00元	1
保 百炼双攻刀【买的放心，用的安心】 有图 信用等级： 物品种类：装备/武器 游戏/区/服：逆冰塞OL/风融竹/凤舞九天	505.00元	1
保 67完美萃力青帝戒【在线发货，方便快捷】 有图 信用等级： 物品种类：装备/戒指 游戏/区/服：逆冰塞OL/将进酒/紫禁之巅	2200.00元	1
保 四攻内功戒指640+【全天在线，即买即发】 有图 信用等级： 物品种类：装备/戒指 游戏/区/服：逆冰塞OL/如梦令/刀剑如梦	2000.00元	1

items and money acquired virtually in games can be sold to other players for real proft in actual currency.

01

Game bot is **harmful**

- Cut down the life span of a game
- Harm the interest of Users

01

Introduction

Game bot is **diverse**

varying with different game conditions and spreading throughout the game universe

01

Introduction

Traditional methods for game bot detection:

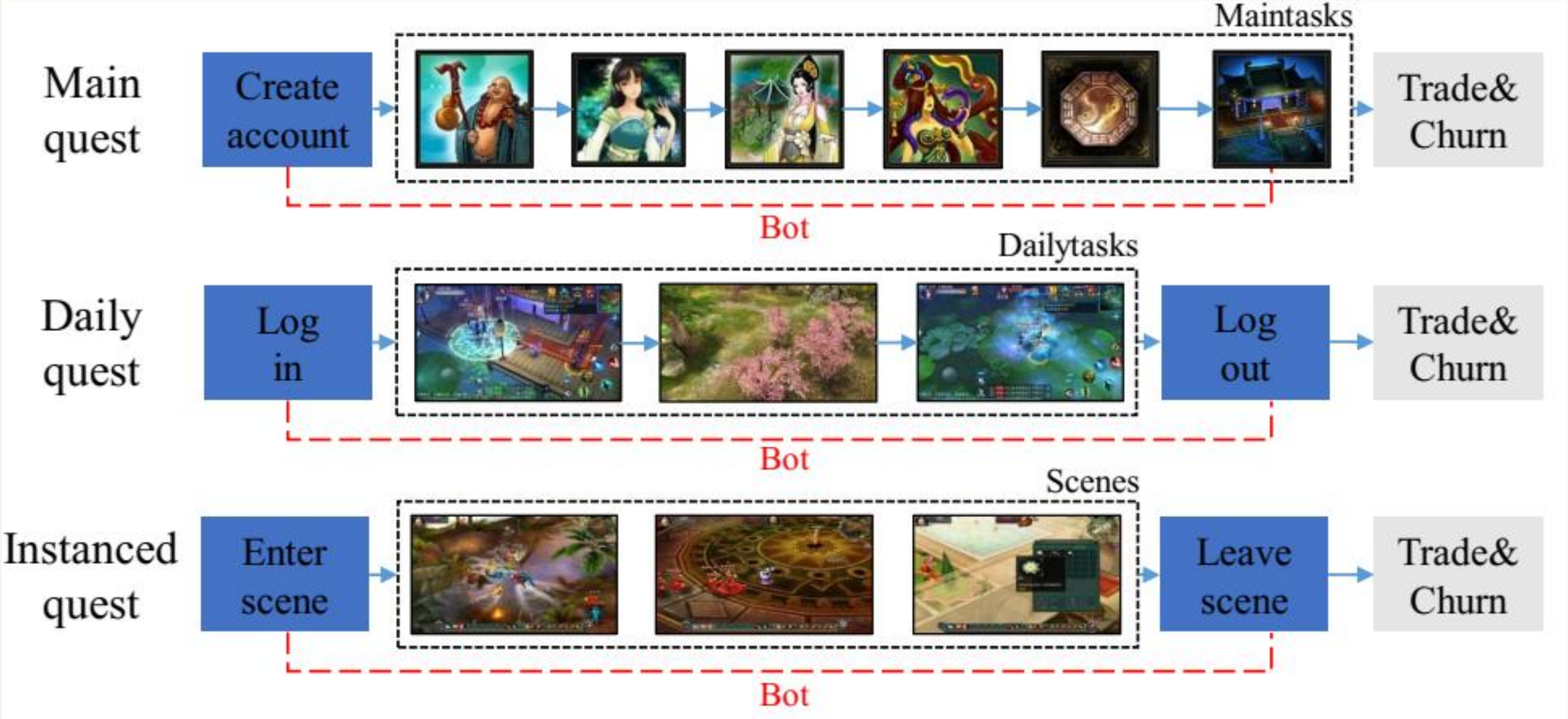
- Handcraft features
- Depend greatly on labeled data
- Can hardly generalize to new games

NGUARD :

- More discriminative features extracted by deep learning
- With the help of unlabeled data
- Can be extended to other games

02

Dataset description



(c) Game bot in MMORPG

02

Dataset description

Each user log is composed of game events ordered by time stamp, which represents each player’ s behavioral sequence

[22:58:20] 使用xxx技能造成xxx伤害
[22:58:20] 获得1134经验
[22:58:23] 获得110金币
[22:58:28] 获得1134经验
[22:58:28] 获得1134经验
[22:58:30] 获得1134经验
[22:58:50] 获得26086经验
[22:58:51] 已通关[xxxx]副本
…

02

Dataset description

Feature Extraction:

- Event ID — A player uses a certain skill, obtains a certain item
- Interval — The time that has passed between the last and the current game event
- Count — The count of times that a certain game event happened during the current sampling time window
- Level — The current game level for the player

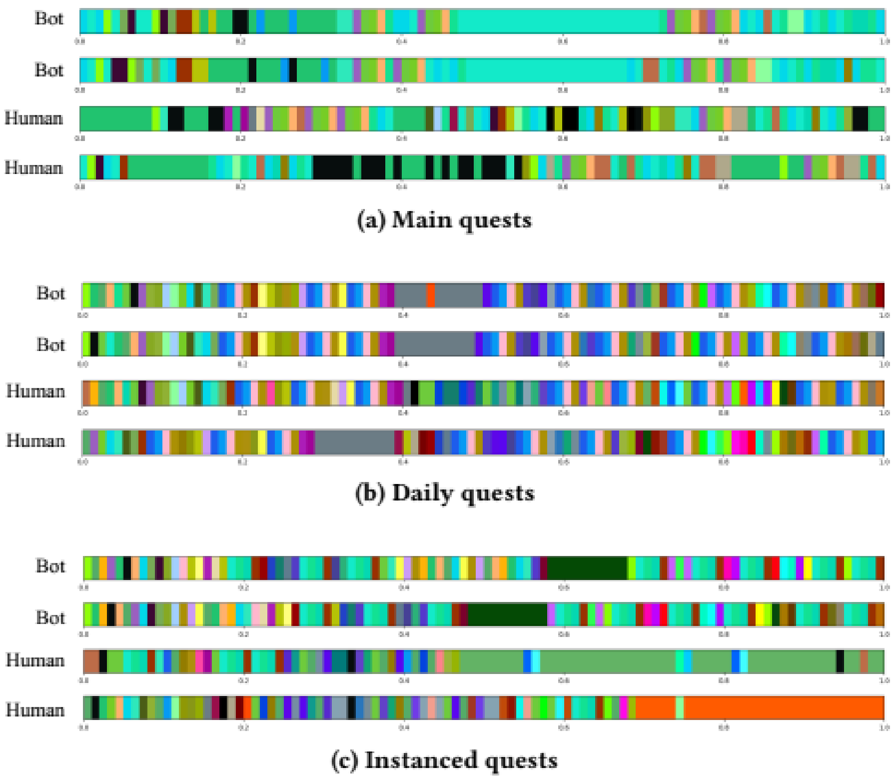


Figure 2: Behavior sequence of game bots is similar to each other, while behavior sequence of human players shows diversity

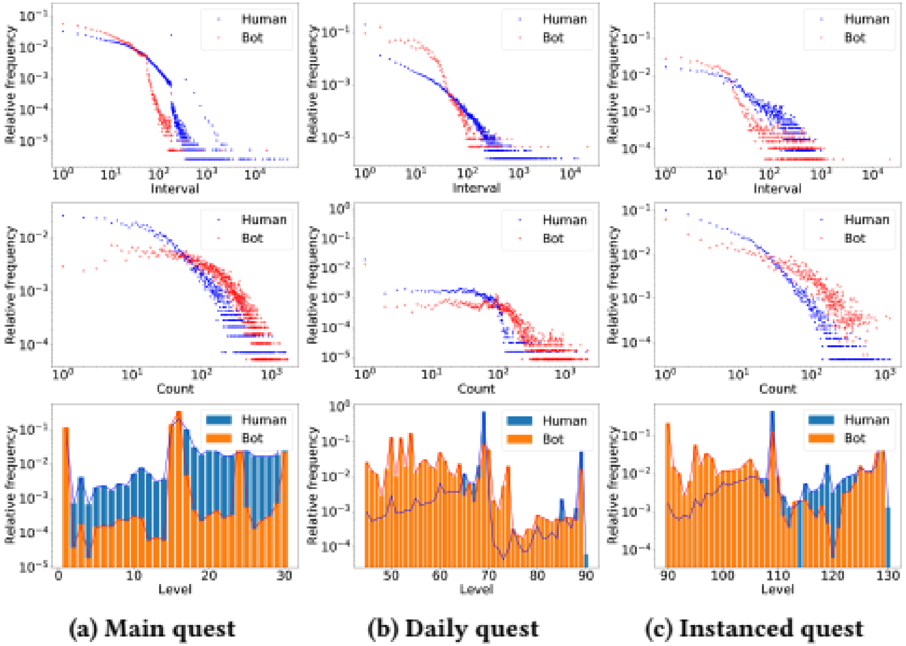


Figure 3: Relationship between features and their relative occurence frequency

Preprocessing:

- Segmentation

Main quest: segment by level
Daily quest: segment by day
Instanced quest: segment by enter scene and leave scene

- Sampling

For data of human players:
Sample from each level interval uniformly

For data of game bots:
according to the density of the original set. Sample more examples in low density area, less for the high density area.

Training data:

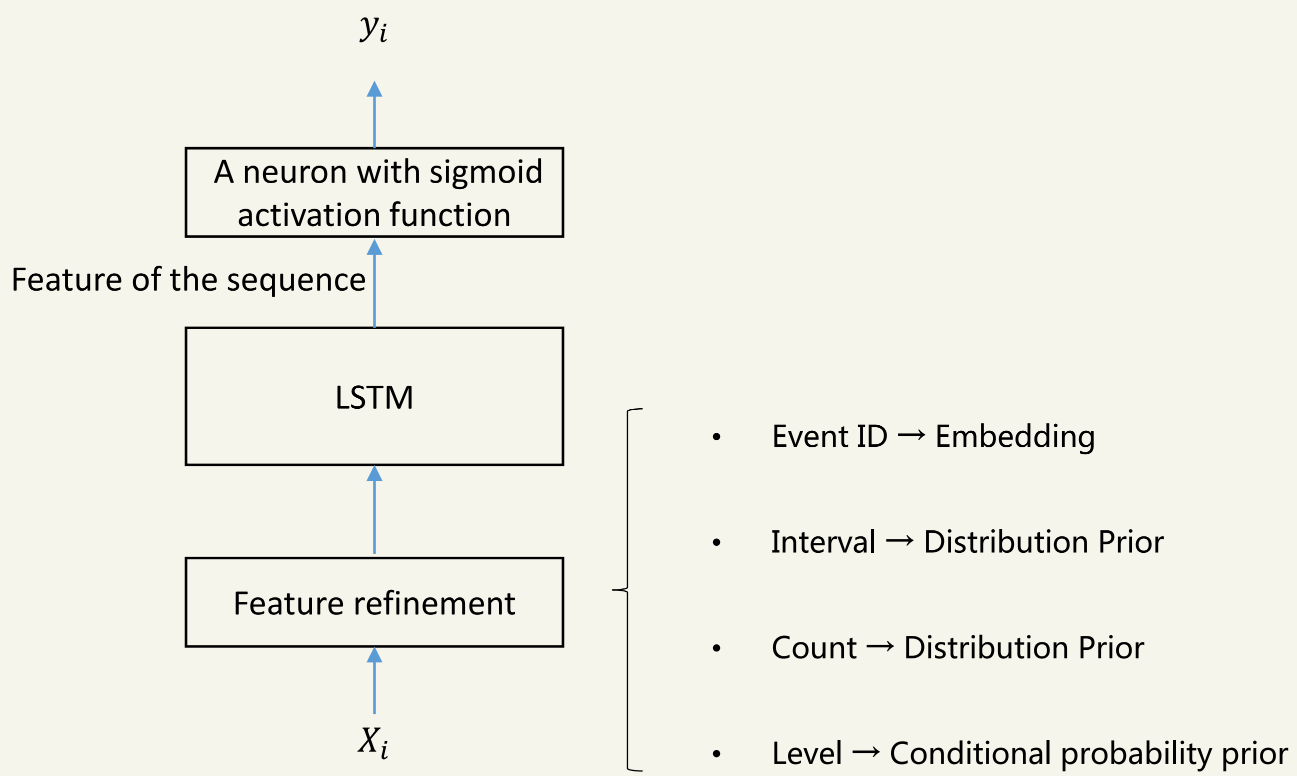
X	y
[(Event13), (Event26), ... (Event113)]	human
[(Event16), (Event39), ... (Event96)]	human
⋮	⋮
[(Event65), (Event32), ... (Event331)]	bot

For each type of quest

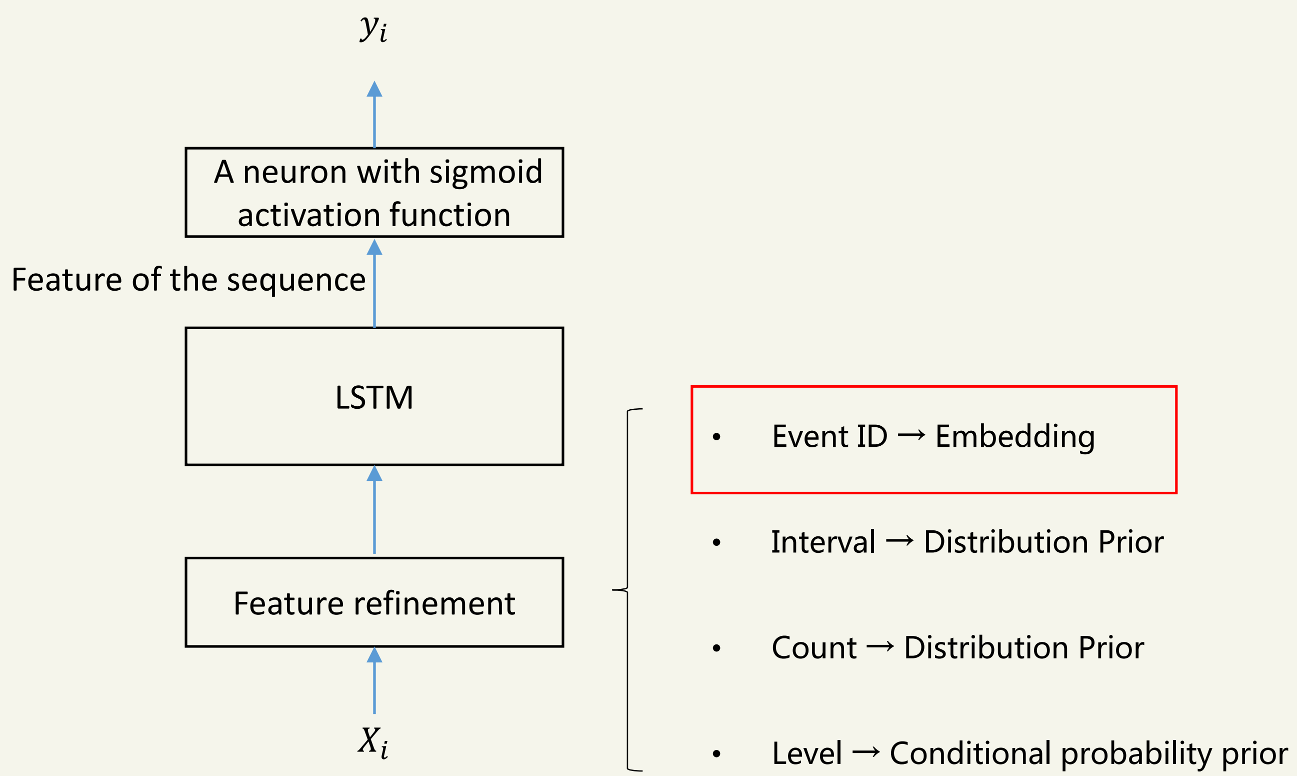
The Framework

Sequence Data?

RNN! But with a quite lot of additional operations …



y_i
A neuron with sigmoid activation function
Feature of the sequence
LSTM
Feature refinement
X_i
• Event ID → Embedding
• Interval → Distribution Prior
• Count → Distribution Prior
• Level → Conditional probability prior



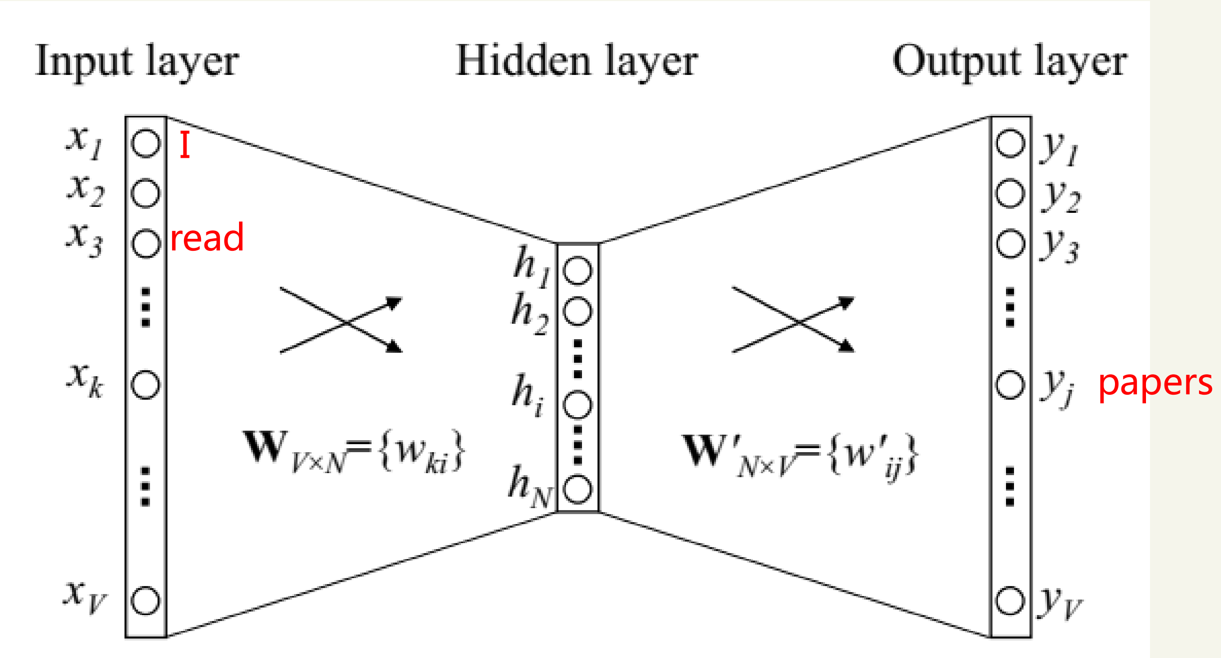
y_i
A neuron with sigmoid activation function
Feature of the sequence
LSTM
Feature refinement
X_i
• Event ID → Embedding
• Interval → Distribution Prior
• Count → Distribution Prior
• Level → Conditional probability prior

NIPS' 13	**Word2Vec:**	(Game bots are automated programs that assist cheating users …… mechanism.)
ASONAM' 16	**APP2Vec:**	((Weibo, 10s), (Weibo, 10s), (Weibo, 5s), (QQ, 20s)…… (Wechat, 0s))
	Event2Vec	((Event1, Interval1), (Event2, Interval2),…… (Eventn, Intervaln))

Word2Vec:

I read papers.

$f('I', 'read') = \text{papers}.$

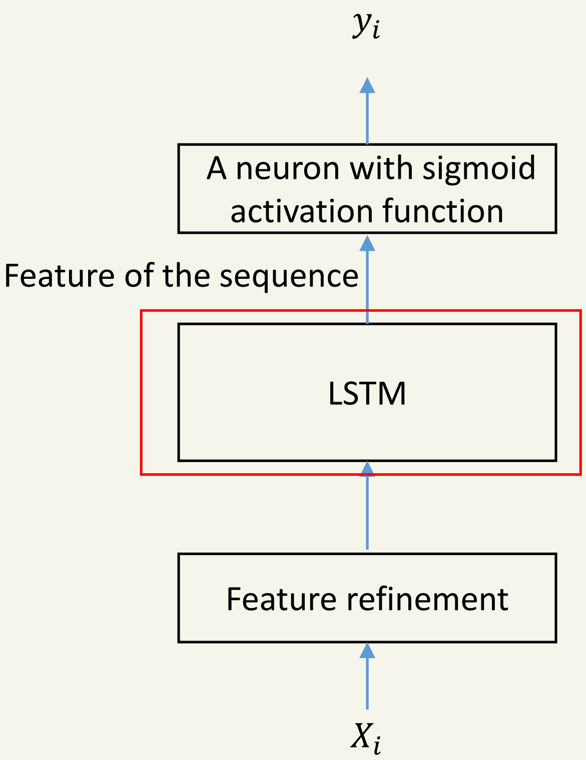


APP2Vec:

((Weibo, 10s), (Weibo, 10s), (Weibo, 20s), (QQ, 15s)...... (Wechat, 0s))

Intuitively the app sessions within short time gaps to the target app should contribute more in predicting the target app.

$$r(w_t, w_c) = 0.8^l$$

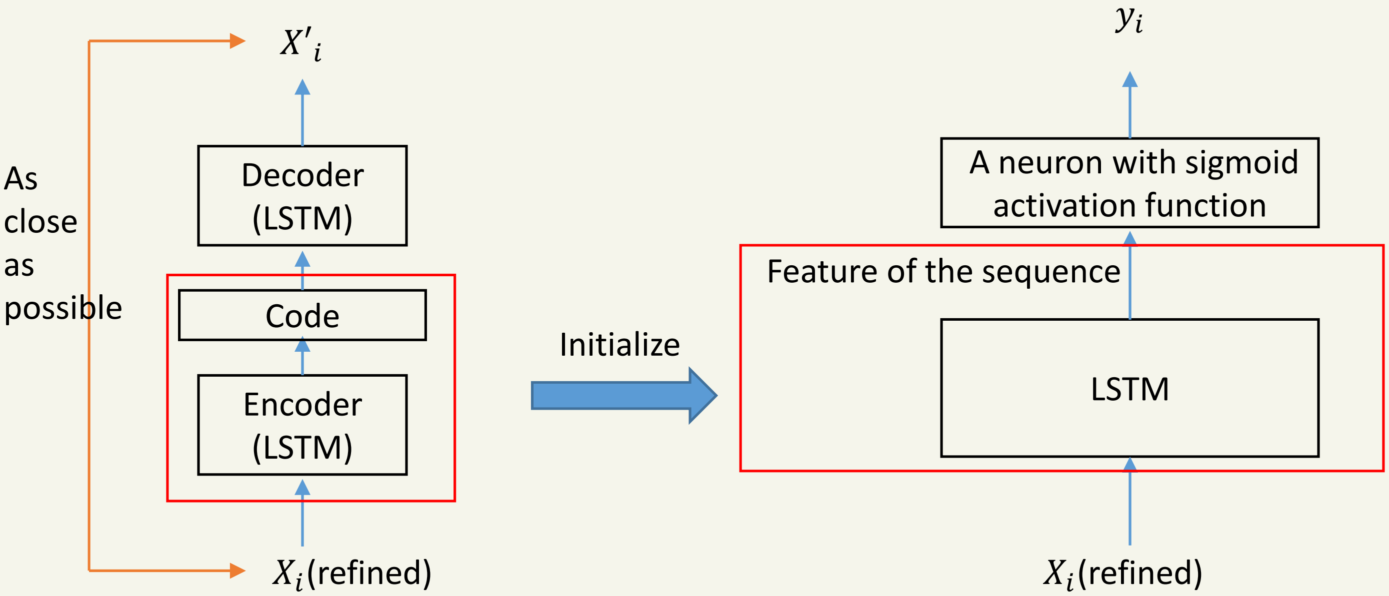


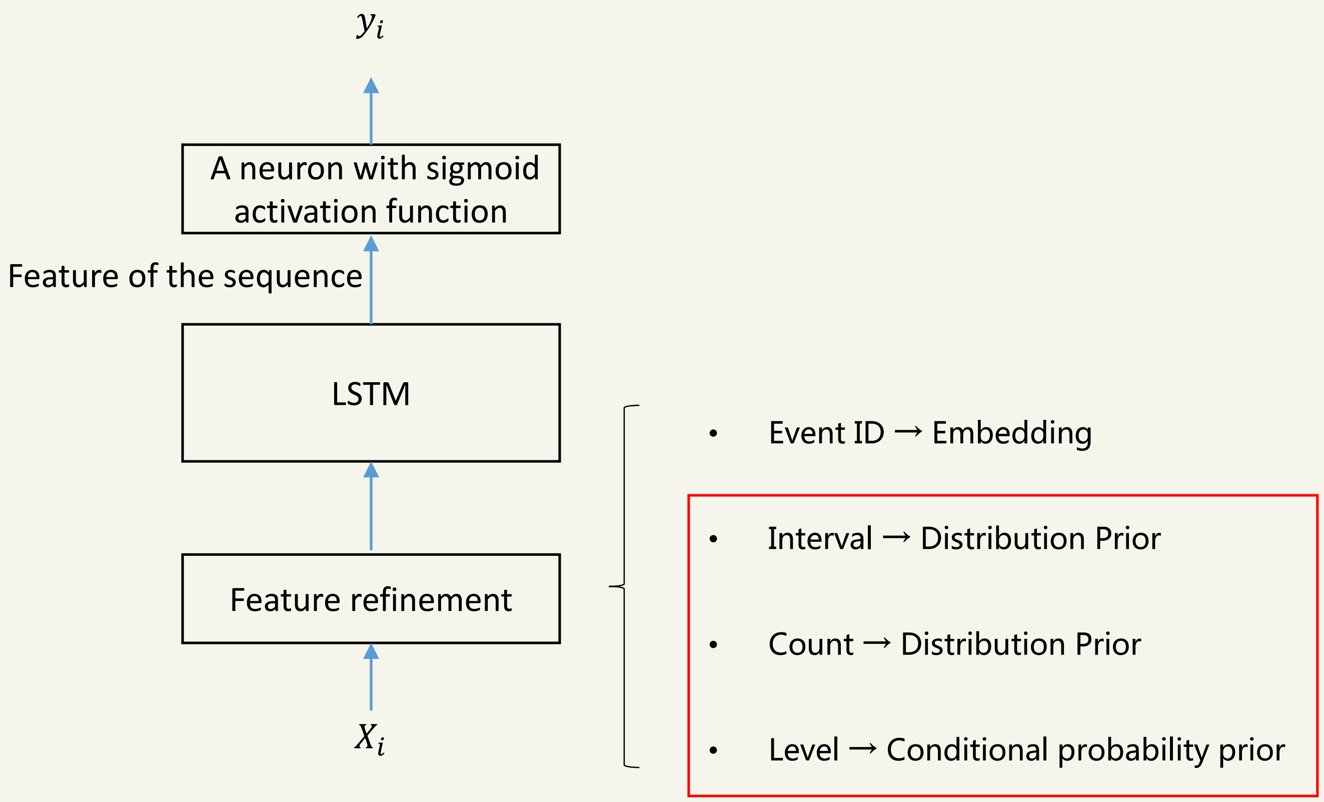
- Event ID → Embedding
- Interval → Distribution Prior
- Count → Distribution Prior
- Level → Conditional probability prior

Sequence Autoencoder and Attention-based Bidirectional LSTM:

For initializing the classification LSTM

For classifying bot




y_i
A neuron with sigmoid activation function
Feature of the sequence
LSTM
Feature refinement
X_i
• Event ID → Embedding
• Interval → Distribution Prior
• Count → Distribution Prior
• Level → Conditional probability prior

Based on authors knowledge of game events, the occurrence of a certain event is a Poisson process.

Interval and Count of each type of event for a human and for a bot fits a gamma distribution respectively.

$$f(x; \alpha, \beta) = \frac{\beta^{\alpha} x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}$$

For the i_{th} event in a sequence, we compute two probabilities:

$$I_{i,h} = f\left(t_i; \alpha_{i,h}, \beta_{i,h}\right) \text{ , } I_{i,b} = f\left(t_i; \alpha_{i,b}, \beta_{i,b}\right) \tag{5}$$

$$C_{i,h} = f\left(c_i; \alpha_{c,h}, \beta_{c,h}\right) \text{ , } C_{i,b} = f\left(c_i; \alpha_{c,b}, \beta_{c,b}\right) \tag{6}$$

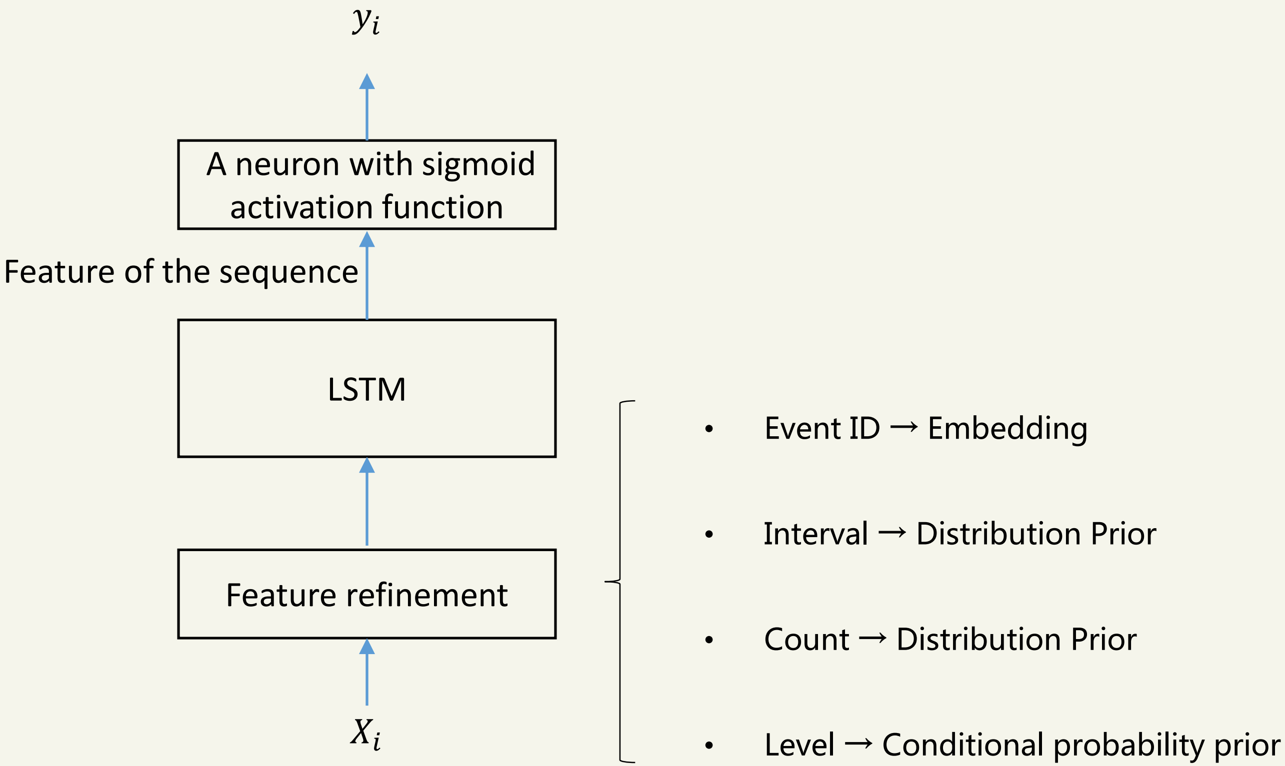
Level:

the occurrence probability of game bots for the current level.

$$\frac{Num\_of\_bots}{Num\_of\_total\_player}$$

Bias:

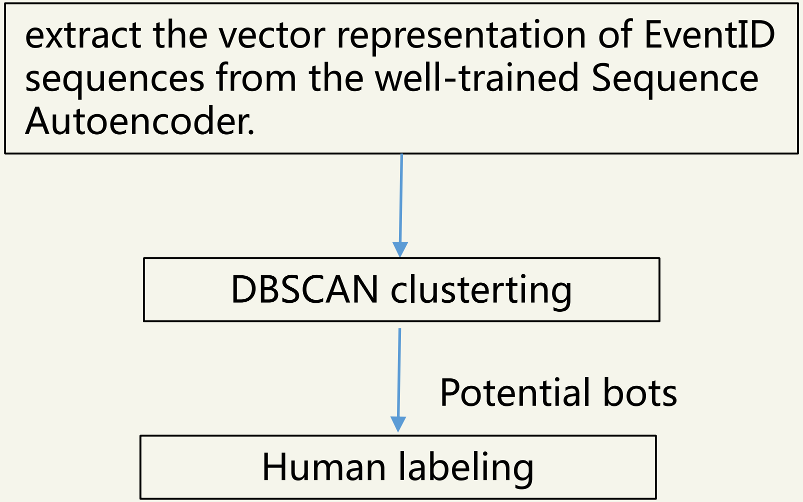
the occurrence probability of game bots for this kind of event.


y_i
A neuron with sigmoid activation function
Feature of the sequence
LSTM
Feature refinement
X_i
Event ID → Embedding
Interval → Distribution Prior
Count → Distribution Prior
Level → Conditional probability prior

Clustering:

Since human players holds the majority in online data, we can easily locate the clusters of human players. And the small clusters surrounding the clusters of human players are different types of game bots.

The mutated bots and unknown bots are hard to be detected by classification model, but easy to be located by clustering

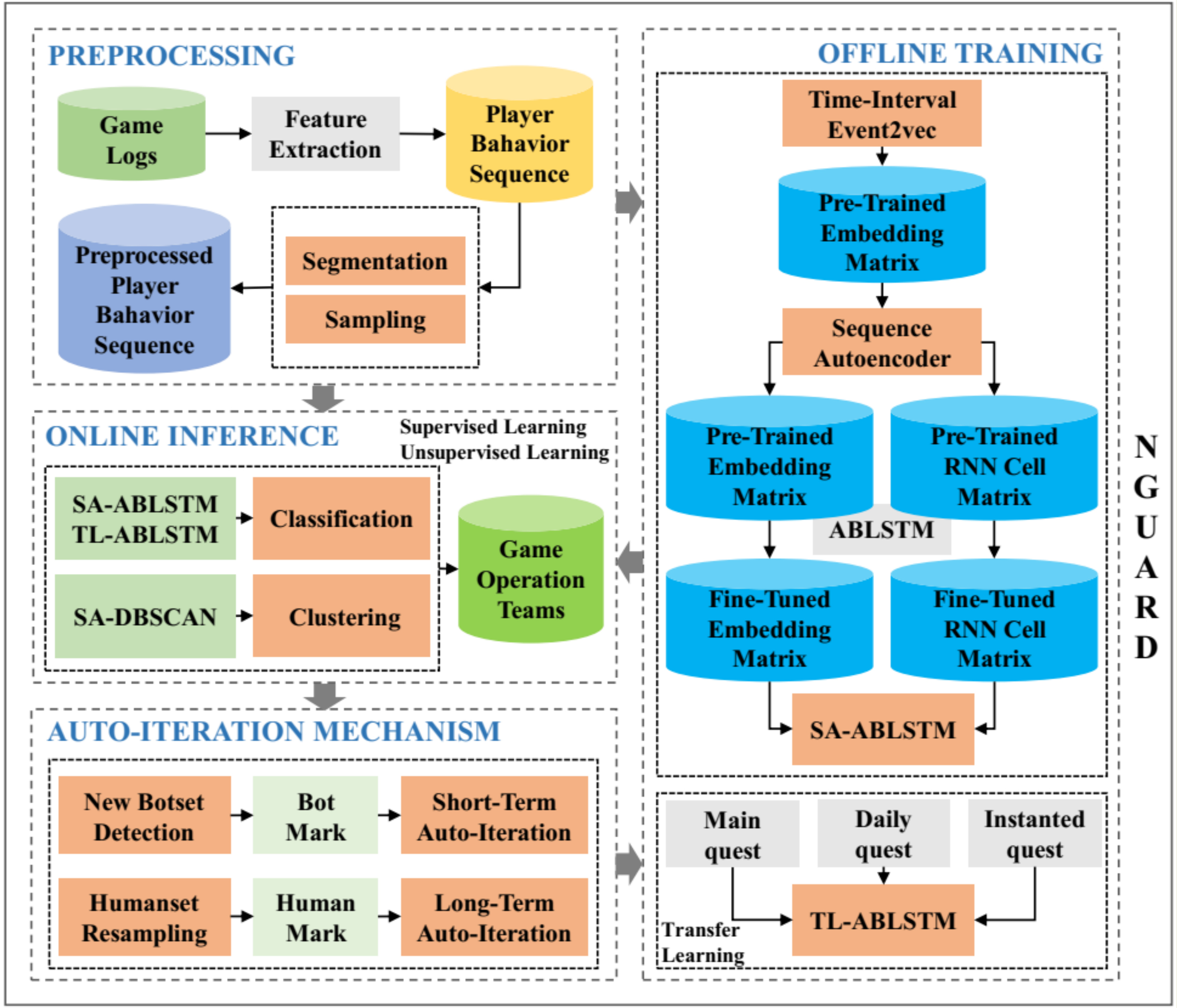

extract the vector representation of EventID sequences from the well-trained Sequence Autoencoder.
DBSCAN clusterting
Potential bots
Human labeling

Incremental-learning:

Game environment changes over time. A model outdated soon after the deployment.

Solution:

Add new samples constantly as the game changes.


PREPROCESSING
Game Logs
Feature Extraction
Player Bahavior Sequence
Preprocessed Player Bahavior Sequence
Segmentation
Sampling
ONLINE INFERENCE
Supervised Learning
Unsupervised Learning
SA-ABLSTM
TL-ABLSTM
Classification
SA-DBSCAN
Clustering
Game Operation Teams
AUTO-ITERATION MECHANISM
New Botset Detection
Bot Mark
Short-Term Auto-Iteration
Humanset Resampling
Human Mark
Long-Term Auto-Iteration
OFFLINE TRAINING
Time-Interval Event2vec
Pre-Trained Embedding Matrix
Sequence Autoencoder
Pre-Trained Embedding Matrix
Pre-Trained RNN Cell Matrix
ABLSTM
Fine-Tuned Embedding Matrix
Fine-Tuned RNN Cell Matrix
SA-ABLSTM
Main quest
Daily quest
Instanted quest
Transfer Learning
TL-ABLSTM
NGUARD

Experiment

Compared methods:

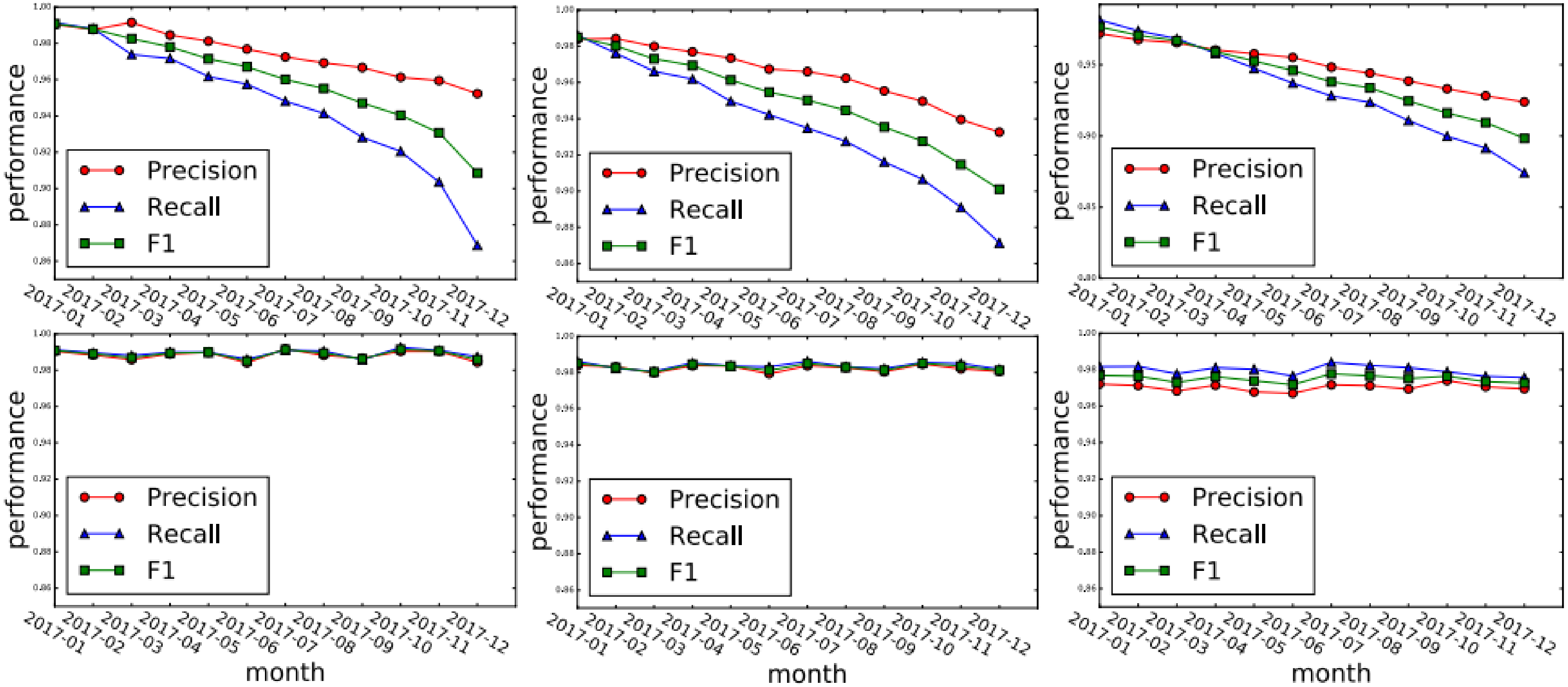
(1) MLP model with 2 fully-connected layers, whose input is the frequencies of EventIDs;

(2) CNN model with 1 convolution layers, followed by average pooling and 1 fully-connected layer;

(3) Bi-LSTM model with 1 layer of Bi-LSTM, following by 1 fullyconnected layer.

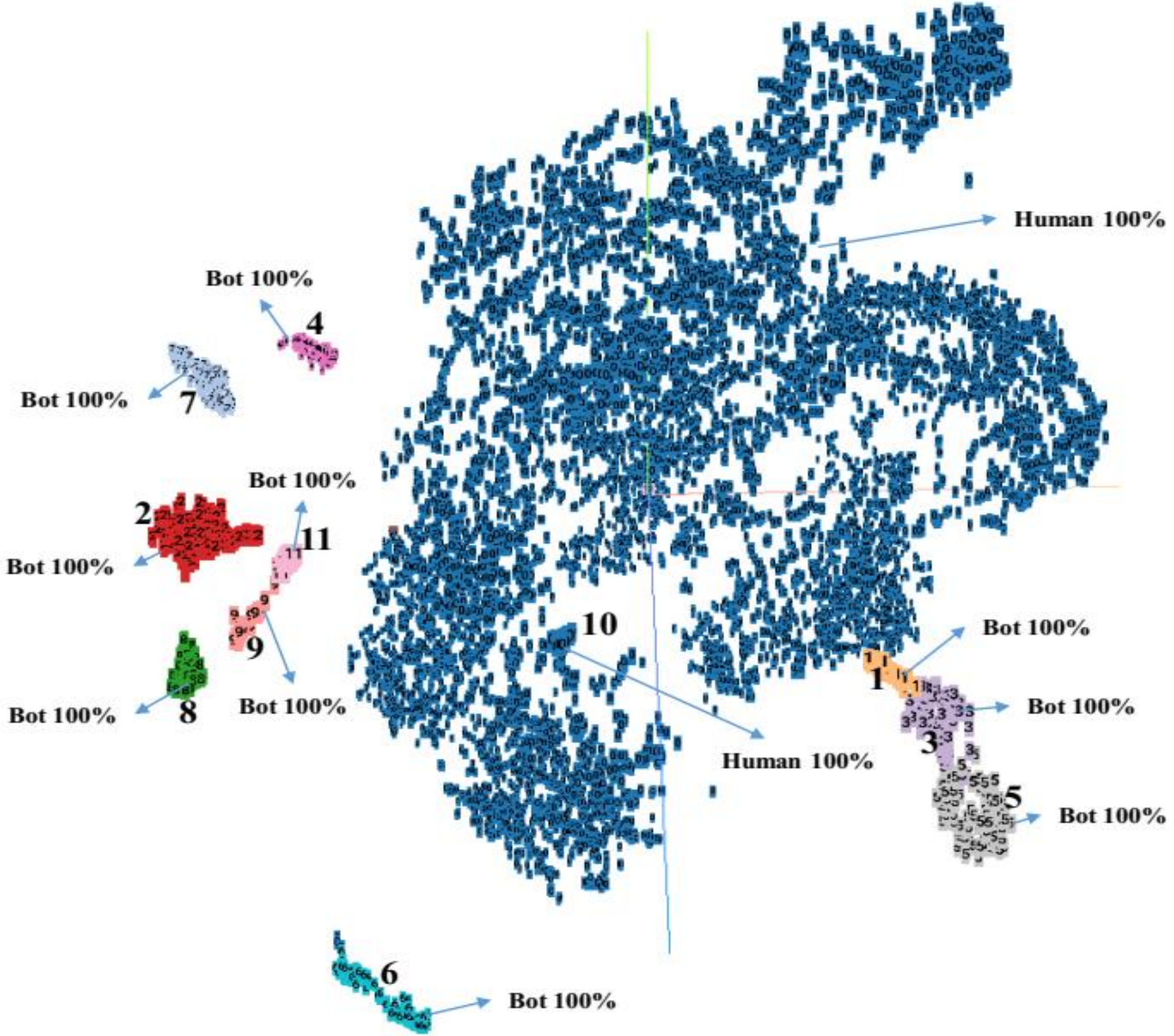
Table 1: Performance comparison of supervised methods

Main quest				Daily quest				Instanced quest			
Model	Precision	Recall	F1	Model	Precision	Recall	F1	Model	Precision	Recall	F1
MLP	0.9618	0.9773	0.9694	MLP	0.9528	0.9609	0.9568	MLP	0.9441	0.9571	0.9506
CNN	0.9721	0.9807	0.9764	CNN	0.9633	0.9712	0.9672	CNN	0.9552	0.9643	0.9597
Bi-LSTM	0.9809	0.9865	0.9837	Bi-LSTM	0.9709	0.9728	0.9718	Bi-LSTM	0.9612	0.9732	0.9672
ABLSTM	0.9851	0.9882	0.9866	ABLSTM	0.9716	0.9774	0.9745	ABLSTM	0.9674	0.9786	0.9730
TL-ABLSTM$_{im}$	0.9878	0.9896	0.9887	TL-ABLSTM$_{id}$	0.9736	0.9721	0.9728	TL-ABLSTM$_{di}$	0.9698	0.9801	0.9749
TL-ABLSTM$_{dm}$	0.9893	0.9906	0.9899	TL-ABLSTM$_{md}$	0.9771	0.9742	0.9756	TL-ABLSTM$_{mi}$	0.9704	0.9808	0.9756
SA-ABLSTM	**0.9904**	**0.9912**	**0.9908**	**SA-ABLSTM**	**0.9838**	**0.9861**	**0.9815**	**SA-ABLSTM**	**0.9721**	**0.9816**	**0.9768**



(a) Main quests (b) Daily quests (c) Instanced quests

Figure 9: Performance with the auto-iteration mechanism



Human 100%
Bot 100%
4
7
Bot 100%
Bot 100%
2
11
Bot 100%
9
10
Bot 100%
8
Bot 100%
1
Bot 100%
3
Human 100%
5
Bot 100%
6
Bot 100%

THANKS