

Final Team Report: A Novel GWAP Disaster Monitoring System

Team *Hotpot*

Changkun Ou^{*}, Yifei Zhan[†] and Zhe Li[‡]

Lecture Human Computation

Institute for Informatics, Ludwig-Maximilian University of Munich, Germany
{changkun.ou^{}, yifei.zhan[†], li.zhe[‡]}@campus.lmu.de*

August 11, 2017

CONTENTS

1	Introduction	3
1.1	Related Works	3
1.1.1	UNICEF Challenges and Satellite Resources	3
1.1.2	Human Computation System and Game With A Purpose	4
1.2	Purpose of the System	4
2	Functionalities	4
2.1	Functionalities as Seen by a Player	4
2.1.1	Disaster Monitoring Game Introduction	4
2.1.2	Game incentivization	5
2.1.3	Example: Interaction Journey	5
2.2	Functionalities as Seen by a Stakeholder	7
2.3	Possible Technology Stack for Implementation	8

3 Design and Models	8
3.1 System Architectures	9
3.2 System Components	10
3.2.1 Database Fields	10
3.2.2 Player Task Generator	12
3.2.3 Player Rating Model	12
3.2.4 Disaster Evaluation Model	19
3.3 Model Initialization	20
4 Evaluation	20
4.1 Success Criterias	21
4.1.1 Model Evaluation by Simulation	21
4.1.2 Issues on Social and Ethical Aspects	22
4.2 Limitations of the System	23
4.2.1 Evaluation Outdated	23
4.2.2 Gameplay and Playability	23
5 Conclusions and Future Works	24
5.1 Conclusions	24
5.2 Future Works	24
Acknowledgements	25
References	25

ABSTRACT Disaster Monitoring is a challenging problem due to the lack of infrastructures in the disaster areas. This report contributes to Disaster Monitoring with the description of a Game With A Purpose, which analyzes tagging results from players from satellite pictures and exports aggregated results to Disaster Monitoring stakeholders. We illustrate our system mockup with proposed implementation technology stack first, then we give the mathematical model of system scheme. As justification and evaluation, we prove the correctness of the model, discuss issues caused by this system and possible solutions extensions for future works as well.

1 INTRODUCTION

In this chapter, we give an introduction about the general backgrounds and challenges of disaster monitoring. Then we briefly introduce the related technics that used in the report, and put forward our general purpose of this report.

1.1 RELATED WORKS

1.1.1 UNICEF CHALLENGES AND SATELLITE RESOURCES

The United Nations Children's Fund (**UNICEF**) [1] is a United Nations programme headquartered in New York City that provides humanitarian and developmental assistance to children and mothers in developing countries. It works in 190 countries and territories to protect the rights of every child. UNICEF has spent 70 years working to improve the lives of children and their families. Defending children's rights throughout their lives requires a global presence, aiming to produce results and understand their effects. For example, in Syria, the UNICEF works on providing and transporting critical medicine, aid and supplies to the refugees living in the war areas [2].

The challenges UNICEF meets is that there are many hard-to-reach and besieged areas (lack of infrastructure), the supplies are extremely difficult to be delivered to these zones if the UNICEF is not aware of the real time war situation and the disaster level. It leads huge costs for the UNICEF which is just a non-profit organization if they would have to hire employees to collect the data of war situation.

With limitations of infrastructure, disaster can only be monitored from the sky level. Satellite sensors may be used to observe the disaster areas, which mainly gives image information of monitoring areas. Zhang et al. [3] discussed the application of a national integrated system using remote sensing, geographic information for monitoring and evaluating flood disaster. Their system has been applied for three years, which gives a proof of the success of satellite sensor informations.

1.1.2 HUMAN COMPUTATION SYSTEM AND GAME WITH A PURPOSE

Human Computation system is a paradigm for utilizing the human processing power to solve problems that computers cannot yet solve [4]. It is the system of computers and large numbers of humans that work together in order to solve problems that could not be solved by either computers or humans alone [5]. **Game With A Purpose (GWAP)** was first proposed in [6, 7], it is a human computation technique of outsourcing steps within a computational process to humans in an entertaining way (gamification). Nevertheless, the data collection mechanisms for a game is variety that should be considered in a proper way [8].

1.2 PURPOSE OF THE SYSTEM

This report sketches a GWAP, aiming at involving citizens to collect data on HtR and BSG [2] areas, which helps organizations like UNICEF. For the collection mechanism of this game, we require game players to select a region upon the presented satellite images and tag the most relevant tags from the provided tags. However, with this gameplay, we need to design carefully to solve issues like malicious player detection, game incentivization, disaster level calculation, etc. In the subsequent chapters, we will discuss the system functionalities first by presenting the system mockups, then gives the comprehensive design of the system backend models for malicious player detection and disaster level calculation as well.

2 FUNCTIONALITIES

In this chapter, we introduce two interactive mockups of our disaster monitoring model. The first mockup is built for game players, which is used for collecting human inputs. The second mockup is built for collaborative organisations, such as [UNICEF](#) and non-governmental organizations (NGOs), who wants to dispatch their rescue teams in disaster area more properly.

2.1 FUNCTIONALITIES AS SEEN BY A PLAYER

2.1.1 DISASTER MONITORING GAME INTRODUCTION

The idea behind our human computation system is [GWAP](#). We have sketched mockups of an image tagging game, which is similar to the ESP game on Artigo [6, 9]. In our game, A player can finish infinity round tasks, a round task contains n image tagging tasks and a tagging task is to: interpret one picture. Within one round task, the player will see n images. Each time he/she will be asked to tag one of these images. At first, the player needs to draw a rectangle area which indicates that he/she has seen some objects in this area. Thoses objects are often considered as a sign of danger or damage. They are mostly like:

- “Rocket Launcher”

- “Armoured vehicles”
- “Tanks”
- “Burning building”
- “Heavy vehicle tracks”
- “sign of explosion”

The system organisation shall predefine a tag list into this system, which is done by a initial trusted group (discuss in section 3.3).

A submenu list (“Pre-provided item list”) which contains all of these items will pop up after the area has been selected. In this step, the player can decide which one of these items matches and then choose it. The player can also obtain helps from the reference panel which provides the player with examples. Besides, if the player does not find the expected item in the “Pre-Provided item list”, he/she can also create a new tag.

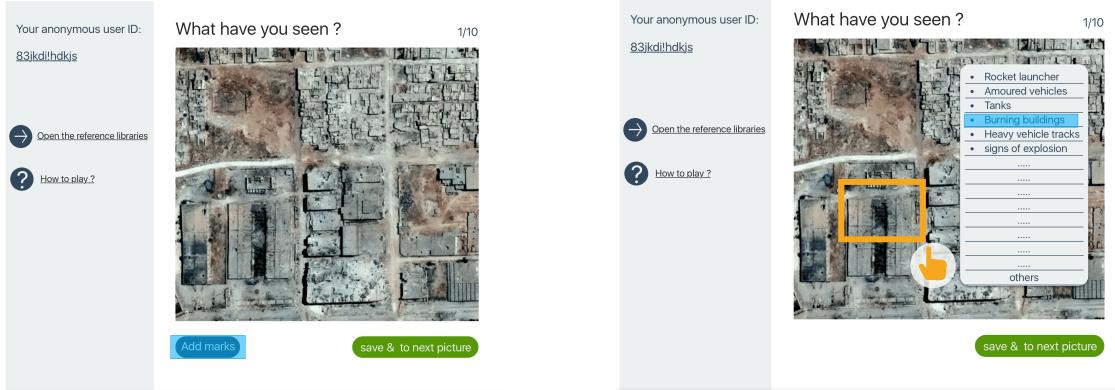
In a tagging task, the player also has a choice to add multiple tags to one picture, or he/she doesn't need to add any tags when he/she thinks that this region is safe. After finishing one tagging task, the player will be directed to the next task till the end of the game.

2.1.2 GAME INCENTIVIZATION

The main incentivization of our Disaster Monitoring Game is altruism. In other words, everyone is willing to help others. Under this consideration and also the consideration of transparency, besides the game guide ,we suggest to provide "acknowledgement page" to the player after he/she finished playing our game. This page contains some information about our research group, the status quo of the disaster region and what is contribution of the player. By offering the player with such a page, we hope our players feel more motivated.

2.1.3 EXAMPLE: INTERACTION JOURNEY

In this section, An example of a user journey is provided in order to illustrate the game process.



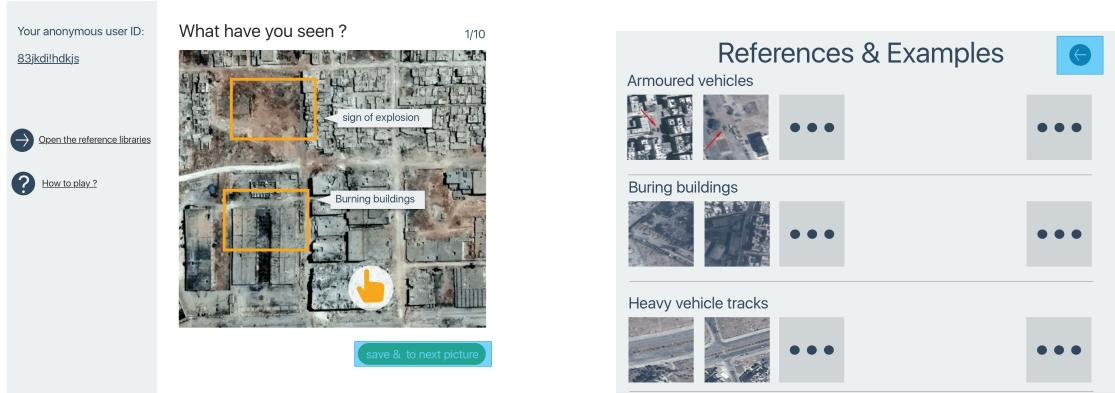
(a) Overview

(b) Tag List

Figure 2.1: Game panel Mockups, satellite image from [10]

Imaging that user A is now at the very first begining of the game. What he will see, are a side bar on the left side, which contains some information like: user ID, game guide and a reference library, and a main panel on the right side (Figure 2.1a). If user A finds a sign of damage or danger in the picture on the right side. He can click the button “add marks” and then hs is able to draw a rectangle area which indicates the location of the sign. A selection box will then pop out automatically and user A can select the suited item or add a new tag (Figure 2.1b).

As already mentioned above, user A can also add multiple tags to one picture (Figure 2.2a).



(a) Multiple Tags

(b) Reference library

Figure 2.2: Game panel Mockups, satellite image from [10]

After user A finishes this tagging task, that is, tagging the first picture, he can click the button “save and to the next picture” so that he can save the tags and go to the next task. Figure 2.2b illustrates how a reference library can be, whose aim is to help user A to identify different

signs of danger or damage. And the acknowledgement page will be provided in the end of the game (Figure 2.3)



Figure 2.3: The acknowledgement page

2.2 FUNCTIONALITIES AS SEEN BY A STAKEHOLDER

In our disaster monitoring system, we take image taggings from players as our input. In the next step, we will filter and analyse the data. The final output of our work is a disaster level report in some certain region, which can be used by some organisations, like: Unicef, some NGOs and even governments.

Under this consideration, we also sketches a mockup for this user group, which gives them an overview of the disaster level and to download the report at the same time (see figure 2.4a).

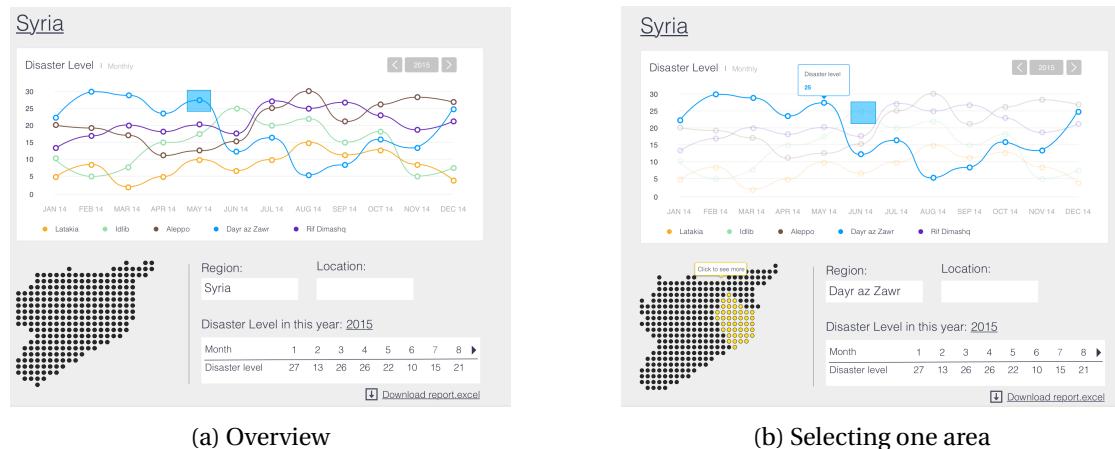


Figure 2.4: Disaster level report platform Mockups

The platform is composed of a curve chart, a map and some statistics. In the curve chart, each

curve shows the the yearly disaster level of a smaller region. Like here in figure 2.4a, the whole region is Syria and each curve stands for a province in Syria. The user can click the “dot” on curves, which stands for the disaster level of that month. Meanwhile, the corresponding area will be highlighted on the map (figure 2.4b).

In the statistic part, the user gets an overview of the yearly disaster level of a region. He/she also have the opportunity to download it. If the user wants to dig deeper, he/she can click on the highlighted area on this map, which will direct him/her to that region (figure 2.5).



Figure 2.5: Disaster level report platform Mockup: Sub-area selection

In figure 2.5, the user is directed to the province Idlib and now each curve stands for a smaller region in this province, in other words, different cities in Idlib.

2.3 POSSIBLE TECHNOLOGY STACK FOR IMPLEMENTATION

Our system should be capable of easily accessing by both Android and iOS users. We suggest to build our system sketched above as a web-based application since Web App is wide-used nowadays and cross any platform, which gives huge benefits for reducing development costs. In addition, we suggest to choose **Polymer**, a Google front-end framework, as our front-end tool and as back-end tool, we suggest to use **Nodejs** as well as **Python**. For our database, we suggest then to choose **MongoDB**, since it is more suitable for a web-application.

3 DESIGN AND MODELS

In this chapter, we describe the overall design of our disaster monitoring backend system in details. Firstly, we propose our system architecture of this disaster monitoring; Then we specify and justify our most critical system components such as databases design, **Player Task Generator (PTG)**, **Player Rating Model (PRM)** as well as **Disaster Evaluation Model (DEM)**.

With these components, model and databases, the disaster monitoring system can handle common problems in human computation system, such as cold start, malicious player detection, etc. It is also expandable, portable and can be easily applied to any other same image selection and tagging based human computation system in different areas.

3.1 SYSTEM ARCHITECTURES

Figure 3.1 illustrate the overall disaster system design. The system databases are composed of two different type of databases. The first database **PlayerDB** combines **TrustedDB** and **UntrustedDB** where persistent the player property and raw tagging inputs whether the overall result is reliable or not. The second database is called **ResultDB** where persistent the reliable player inputs.

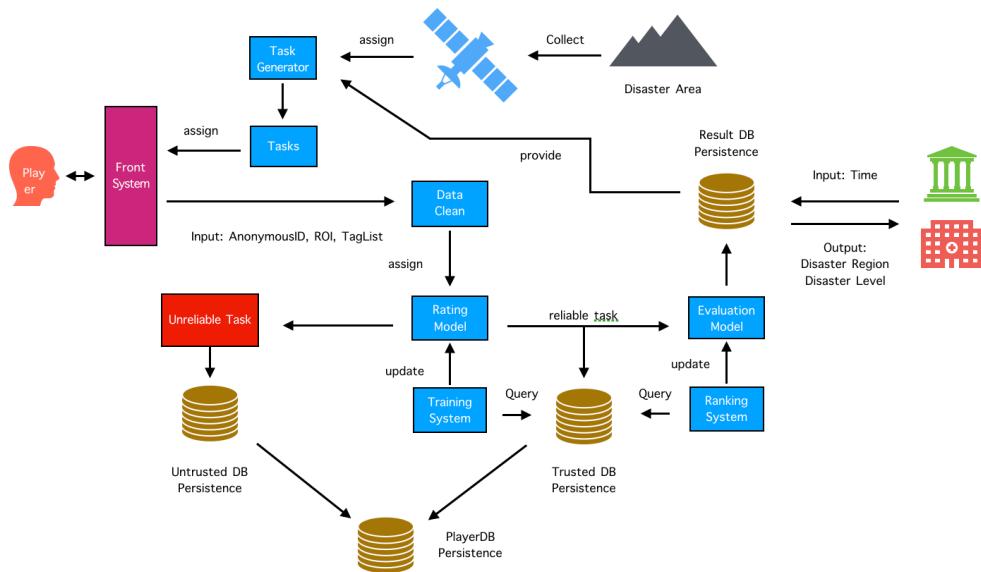


Figure 3.1: System Design Overview

For this architecture design, one can simplify the overall data flow into tree main steps that describes as follows:

- Step 1. Player task generating: We propose the **PTG** that combines trusted results from TrustedDB, and separate new images from satellite, then assign this binding to the future players.
- Step 2. Malicious player detection: A reliable player shall pass the malicious detection algorithm (describe in algorithm 1) inside the **PRM**. Once the player is not a malicious player, then system will mark all the results from this player as a reliable result and then send it into next step.
- Step 3. Evaluating disaster level: the system reuse the reliable player inputs into **DEM** and calculate the disaster level of the monitoring region as well as persistent it in the second

database ResultDB.

After these three main steps, stakeholders are able to retrieve monitoring results from the database ResultDB.

3.2 SYSTEM COMPONENTS

3.2.1 DATABASE FIELDS

We describe the system database PlayerDB fields as well as the fields of database ResultDB first in listing 3.1a and 3.1b.

```
1 [
2   {
3     "player_id": "E3A6F124-4A6C
4       -4C6E-B7F1-F8BC9A7381CC",
5     "tasks": [
6       {
7         "image_id": "3A21E99E-
8           F074-454B-A590-8
9             D8C5ABD8E77",
10        "image_at": "2017-07-31
11          11:28:40",
12        "reliable": true,
13        "ROIs": [
14          {
15            "x": 103, "y": 121,
16            "height": 56,
17            "width": 78,
18            "tags": ["burning
19              building", "explosion"]
20          }
21        ]
22      }
23    ]
24  }
```

```
1 [
2   {
3     "region_id": "FBEB6204-0B94
4       -4811-94F0-9DDC5FBBE6D8",
5     "history": [
6       {
7         "image_id": "3A21E99E-
8           F074-454B-A590-8
9             D8C5ABD8E77",
10        "image_at": "2017-07-31
11          11:28:40",
12        "ROIs": [
13          {
14            "x": 103, "y": 121,
15            "height": 56,
16            "width": 78,
17            "tags": ["burning
18              building", "explosion"]
19          }
20        ]
21      }
22    ]
23  }
```

a: Example of PlayerDB Data

b: Example of ResultsDB Data

Listing 3.1: Examples of Database Fields

In this disaster monitoring system, our participants **do not need to register accounts**, and the system backend shall **generate and assign an *player_id* to each player** according to the

user scenario (such as IP address, network status, system information et cetera). This function significantly accelerate player to participate in this game. Thus, the **PlayerDB** stores the *player_id* to detect the same players if they participate next time. The player will accomplish different game tasks; each task result shall store in the tasks filed.

In the **ResultDB**, a *region_id* is unique and assigned by our system. A region has its monitoring history and composed by its separate images, the only difference between PlayerDB and ResultDB is ResultDB only stores reliable data (no *reliable* field) and images organized by its related region.

To explain other fields and establish our models, we describe few basic definitions for the system models first.

Definition 3.1. *The Region of Interests (ROI) is an indicator that represents the one of the selected two dimensional regions from player. The i-th ROI from player p in image k at image creation time t is denoted by $ROI_{p,i,k,t}$.*

Considering image *k* **implies** its creation time *t* (an image always has its creation time), for convenience, we usually **simplify** $ROI_{p,i,k,t}$ to $ROI_{p,i,k}$. For instance, figure 3.2 shows few examples of ROI on different images.

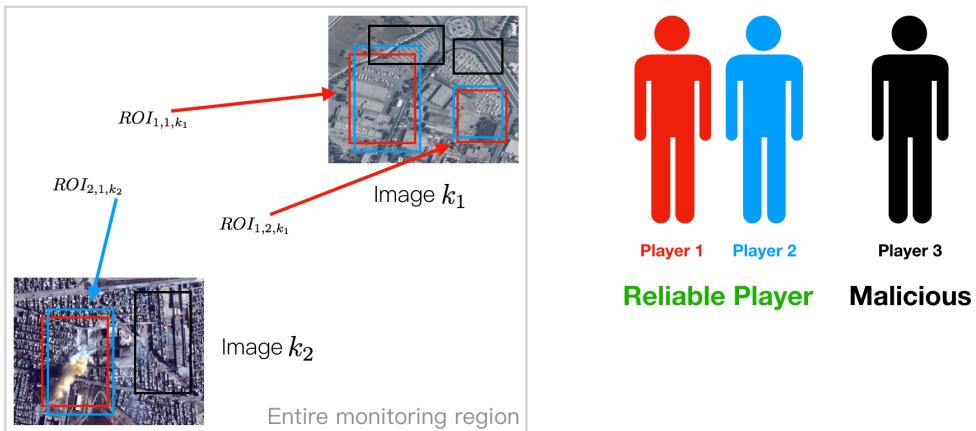


Figure 3.2: Examples of Region of Interests (ROI)

With this definition, our system players are able to select ROIs for each image as well as capable of select tags for each ROI. Thus, the *tasks* field in PlayerDB is an array object, stores each player image result with an assigned *image_id*. Each object in the *tasks* array has a field *reliable*, which indicates the reliability for this object task; Each object also contains a *ROIs* field, which is an array object that contains the player inputs for this object image; Each ROI object in the *ROIs* field has four properties that describes the ROI geometric location: *x*, *y*, *height*, *width*, and also a *tags* array field that describes the input tags for this image from this player.

For *tags* field, game players can select the related tags for each ROI, and stores in this array.

However, systems like ESP [11], ARTigo [9] have proved that human inputs are valuable and useful. Therefore we allows player input their own tags. That is, **each tag can only selects once, and players allow to input their own new tags for the selected ROIs**. Then, We define the ROI tag vector for the model design:

Definition 3.2. *Assuming the database stores n different tags $\text{tag}_1, \text{tag}_2, \dots, \text{tag}_n$ for a certain image k , the Tag Vector $T_{p,i,k}$ of $ROI_{p,i,k}$ (the i -th ROI in image k of player p) is a vector that is donated by the following formula:*

$$T_{p,i,k} = (|\text{tag}_1|, |\text{tag}_2|, \dots, |\text{tag}_n|) \quad (3.1)$$

where

- tag_i is the i -th tag;
- n is the number of tags;
- $|\text{tag}_i|$ is the count of tag_i in a player task object.

Note that due to each tag can only selects once, the components of tag vector is **either 1 or 0** in reality. This definition performs a popular data preprocessing technique, which called One-Hot Encoding trick [12, 13]. For instance, for a certain image k , there are 5 different tags $\text{tag}_1, \text{tag}_2, \text{tag}_3, \text{tag}_4, \text{tag}_5$ were inputed by our game player. Assuming player p selects the first ROI and inputs tags for $ROI_{p,1,k}$: $\{\text{tag}_1, \text{tag}_2, \text{tag}_3, \text{tag}_4\}$, player q selects the first ROI and inputs tags for $ROI_{q,1,k}$: $\{\text{tag}_1, \text{tag}_3, \text{tag}_4, \text{tag}_5\}$. Then tag vector $T_{p,1,k}$ of $ROI_{p,1,k}$ is $(1, 1, 1, 1, 0)$ and tag vector $T_{q,1,k}$ of $ROI_{q,1,k}$ is $(1, 0, 1, 1, 1)$.

3.2.2 PLAYER TASK GENERATOR

The **PTG** combines task images from satellite and ResultDB. A player task contains $2n$ different images in random order, that n images are the untagged new satellite images and n images are tagged images of ResultDB, which means **PTG** contains two generating steps:

- Step 1. **PTG** shall split a monitoring region into small pieces of images, and also assign a unique **image_id** for each piece (The reason is discussed in section 4.1.2 for the leakage of data).
- Step 2. **PTG** shall retrieve tagged images from ResultDB. Then combine all images as a user task assign to a new upcoming player.

3.2.3 PLAYER RATING MODEL

This subsection describes the **PRM** as well as its detection algorithm inside our Disaster Monitoring system. The **PRM** is responsible for detect malicious players regarding their game task inputs.

PageRank was first proposed by Larry Page [14] and applied to social analysis in [15]. It is commonly used for expressing the stability of physical systems and the relative importance, so-called **centralities**, of the nodes of a network. We transfer the basic idea of centralities of a network and use eigenvalue as the trust value for each player to distinguish malicious players and reliable players.

We establish the model in image dependent perspective. For a certain image k , considering a directed **Player Rating Graph (PRG)** between players who tagged the image k . Each player is a node of PRG, as illustrated in figure 3.3.

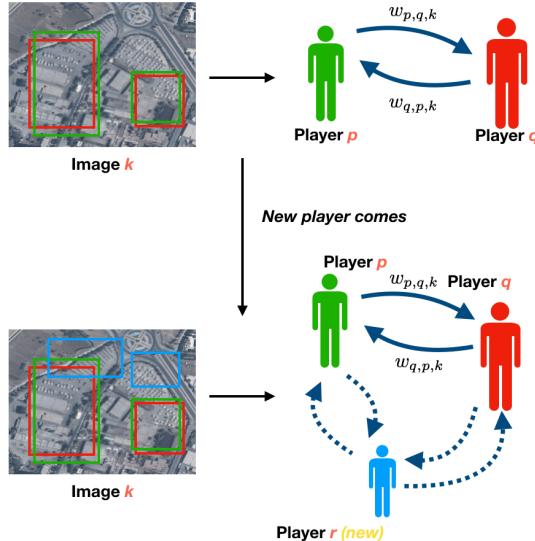


Figure 3.3: Player Rating Graph for Certain Images

Definition 3.3. Assuming the database stores n different tags $\text{tag}_1, \text{tag}_2, \dots, \text{tag}_n$ in the system, The **System Weight Vector** $v = (p(\text{tag}_1), p(\text{tag}_2), \dots, p(\text{tag}_n))$ of all tags can be calculated by the following Equation 3.2:

$$p(\text{tag}_i) = \frac{|\text{tag}_i|}{\sum_{j=1}^n |\text{tag}_j|} \quad (3.2)$$

where $|\text{tag}_i|$ is the count of tag_i in the system.

Definition 3.4. Assuming there are r_s different tags $\text{tag}_{r_1}, \text{tag}_{r_2}, \dots, \text{tag}_{r_s}$ were tagged in a certain image k , the **Image Weight Vector** is a vector for image k that is composed by part of the System Weight Vector, which is donated by $v_k = (p(\text{tag}_{r_1}), p(\text{tag}_{r_2}), \dots, p(\text{tag}_{r_s}))$ with $r_i (i = 1, 2, \dots, s) \in \{1, 2, \dots, n\}$, $r_i \neq r_j (i \neq j, j = 1, 2, \dots, s)$ and $s \leq n$.

For instance, the system have 2 different images. The first image is tagged by two players. One is $\text{tag}_1, \text{tag}_2, \text{tag}_5$ and another is $\text{tag}_1, \text{tag}_2$; The second image is tagged by three players, their results are: $\text{tag}_1, \text{tag}_2, \text{tag}_5; \text{tag}_2, \text{tag}_4, \text{tag}_5; \text{tag}_3, \text{tag}_4, \text{tag}_5$. Thus, the system currently have 5 different tags $\text{tag}_1, \text{tag}_2, \text{tag}_3, \text{tag}_4, \text{tag}_5$. Each tags (tag_1 to tag_5) have corresponding counts:

$3, 4, 1, 2, 4$; Therefore the System Weight Vector is $(\frac{3}{14}, \frac{2}{7}, \frac{1}{14}, \frac{1}{7}, \frac{2}{7})$; the Image Weight Vector of the first image is $(\frac{3}{14}, \frac{2}{7}, \frac{2}{7})$ since the first image only is tagged by $\text{tag}_1, \text{tag}_2, \text{tag}_5$, and the Image Weight Vector of the second image is as same as the System Weight Vector due to the second image is tagged by all exist tags.

Obviously, we have $0 \leq p(\text{tag}_i) \leq 1$, $\sum_{i=1}^n p(\text{tag}_i) = 1$ and $\sum_{i=1}^s p(\text{tag}_{r_i}) \leq 1$.

To go so far as to this system, our player have two different type of inputs: **the ROI, and its Tag Vector**. To define the PRG edge weight, we introduce two input measurements in the subsequent Definition 3.5 and 3.6.

Definition 3.5. *The Players ROI Matching Ratio (PRMR) is an importance measurement that measures the proportion of two different ROI intersection surface from player p, q and the ROI surface from player p in a certain image k , which is donated by the following formula:*

$$\text{PRMR}(p, q, i, j, k) = \frac{|ROI_{p,i,k} \cap ROI_{q,j,k}|}{|ROI_{p,i,k}|} \quad (3.3)$$

where

- $ROI_{p,i,k}$ is the i -th selected ROI from player p ;
- $|ROI_{p,i,k}|$ is the surface of $ROI_{p,i,k}$;

Lemma 3.1.

$$0 \leq \text{PRMR}(p, q, i, j, k) \leq 1 \quad (3.4)$$

Proof. According to the Definition of ROI, $|ROI_{p,i,k} \cap ROI_{q,j,k}|$ can archive its maximum value only and only if $ROI_{p,i,k} = ROI_{q,j,k}$ as well as its minimum value only and only if $ROI_{p,i,k}$ has no intersection with $ROI_{q,j,k}$. Thus:

$$0 = \frac{0}{|ROI_{p,i,k}|} \leq \text{PRMR}(p, q, i, j, k) \leq \frac{|ROI_{p,i,k} \cap ROI_{p,i,k}|}{|ROI_{p,i,k}|} = \frac{|ROI_{p,i,k}|}{|ROI_{p,i,k}|} = 1.$$

□

Definition 3.6. *The Players Input Tag Correlation (PITC) is an importance measurement that measures the proportion of the covariance of two different Tag Vector $T_{p,i,k}, T_{q,j,k}$ from player p, q and the covariance of $T_{p,i,k}$ from player p with itself under the Image Weight Vector v_k , which is donated by the following formula:*

$$\text{PITC}(p, q, i, j, k) = \frac{\text{Cov}(T_{p,i,k}, T_{q,j,k}; v_k)}{\text{Cov}(T_{p,i,k}, T_{p,i,k}; v_k)} \quad (3.5)$$

where $\text{Cov}(X, Y; w)$ is the weighted covariance between X and Y , which donated by:

$$\text{Cov}(X, Y; w) = \frac{\sum_{i=1}^n w_i (x_i - \frac{1}{n} \sum_{i=1}^n w_i x_i)(y_i - \frac{1}{n} \sum_{i=1}^n w_i y_i)}{\sum_{i=1}^n w_i} \quad (3.6)$$

with $X = (x_1, x_2, \dots, x_n)$, $Y = (y_1, y_2, \dots, y_n)$, $w = (w_1, w_2, \dots, w_n)$.

Note that

1. **The definition of PRMR and PITC share the same intent for measuring asymmetric importance between player p and player q (how p think of q);**
2. The definition of **PRMR is inspired by** a wide-used computer vision criteria, the so called **Intersection over Union (IoU)**, also known as **Jaccard Index** [16, 17], which is a statistic used for comparing the similarity and diversity of sample sets. However, in our case, we only divided by to guarantee the asymmetric property for directed graph weight;
3. The definition of **PITC is inspired by the Weighted Pearson Correlation Coefficient** [18], which is a measure of the linear correlation between two variables. In our case, with the same intent of PRMR, we **drop** the part of covariance of player q in denominator to guarantee the asymmetric property for directed graph weight;
4. The PRMR and PITC both **are not metrics** due to $\text{PRMR}(p, q, i, j, k) \neq \text{PRMR}(q, p, i, j, k)$ as well as $\text{PITC}(p, q, i, j, k) \neq \text{PITC}(q, p, i, j, k)$.

Lemma 3.2.

$$-1 \leq \text{PITC}(p, q, i, j, k) \leq 1. \quad (3.7)$$

Proof. We know that the weighted Pearson Correlation Coefficient [18] lies on $[-1, 1]$, i.e.

$$-1 \leq \frac{\text{Cov}(T_{p,i,k}, T_{q,j,k}; v_k)}{\sqrt{\text{Cov}(T_{p,i,k}, T_{p,i,k}; v_k)\text{Cov}(T_{q,j,k}, T_{q,j,k}; v_k)}} \leq 1$$

To prove Equation 3.7, we have to show:

$$\begin{aligned} & \frac{\text{Cov}(T_{p,i,k}, T_{q,j,k}; v_k)}{\text{Cov}(T_{p,i,k}, T_{p,i,k}; v_k)} \\ & \leq |\text{Cov}(T_{q,j,k}, T_{q,j,k}; v_k)|\sqrt{\text{Cov}(T_{p,i,k}, T_{p,i,k}; v_k)\text{Cov}(T_{q,j,k}, T_{q,j,k}; v_k)} \leq 1 \end{aligned} \quad (3.8)$$

and

$$\begin{aligned} & \frac{\text{Cov}(T_{p,i,k}, T_{q,j,k}; v_k)}{\text{Cov}(T_{p,i,k}, T_{p,i,k}; v_k)} \\ & \geq -|\text{Cov}(T_{q,j,k}, T_{q,j,k}; v_k)|\sqrt{\text{Cov}(T_{p,i,k}, T_{p,i,k}; v_k)\text{Cov}(T_{q,j,k}, T_{q,j,k}; v_k)} \geq -1 \end{aligned} \quad (3.9)$$

Then we need to show:

$$0 \leq \text{Cov}(T_{p,i,k}, T_{p,i,k}; v_k)\text{Cov}(T_{q,j,k}, T_{q,j,k}; v_k)^3 \leq 1 \quad (3.10)$$

Considering $T_{p,i,k}, T_{q,j,k}$ are described in general, with Equation 3.6, we only need to show (s is an vector components index instead of exponential):

$$0 \leq \text{Cov}(T_{p,i,k}, T_{p,i,k}; v_k) = \frac{\sum_{s=1}^n v_k^s \left(T_{p,i,k}^s - \frac{1}{n} \sum_{s=1}^n v_k^s T_{p,i,k}^s \right)^2}{\sum_{s=1}^n v_k^s} \leq 1 \quad (3.11)$$

According to the definition of **Tag Vector** and **Image Weight Vector**, the components of $T_{p,i,k}$ are either 1 or 0, the components of v_k lies on $[0, 1]$, then we have:

$$0 \leq \left(T_{p,i,k}^s - \frac{1}{n} \sum_{s=1}^n v_k^s T_{p,i,k}^s \right)^2 \leq 1 \quad (3.12)$$

Therefore,

$$0 = \frac{\sum_{s=1}^n v_k^s \cdot 0}{\sum_{s=1}^n v_k^s} \leq Cov(T_{p,i,k}, T_{p,i,k}; v_k) \leq \frac{\sum_{s=1}^n v_k^s \cdot 1}{\sum_{s=1}^n v_k^s} = 1 \quad (3.13)$$

which proves Equation 3.11. \square

Thus far, we have enough techniques to define the edge weight of **PRG**. The definition is shown in Definition 3.7.

Definition 3.7. *For a certain image k , the edge weight of the PRG from player p to player q is denoted by the formula 3.14:*

$$w_{p,q,k} = \sum_{j=1}^n \sum_{i=1}^m (PRMR(p, q, i, j, k) (PITC(p, q, i, j, k) + 2)) \quad (3.14)$$

with player p selected m ROIs, player q selected n ROIs.

Our goal is to calculate the centrality of the nodes (players) of the network graph. The **Perron Frobenius theorem**, proved by Oskar Perron (1907) [19] and Georg Frobenius (1912) [20], guarantees our goal can be drifted to the calculation of the adjacency matrix of **PRG**. In consequence, one can use the normalized adjacency matrix through the following formula 3.15:

$$A_k = (a_{p,q,k}) = \left(\frac{w_{p,q,k}}{\sum_q w_{p,q,k}} \right) \quad (3.15)$$

where k is the image indicator.

Theorem 3.3. *The normalized adjacency matrix A_k of PRG of a certain image k is irreducible, real, non-negative, column-stochastic, and diagonal element being positive.*

Proof. Irreducibility: As shown in figure 3.3, for a certain image k , the **PRG** is strong connected because the player who selected ROIs in image k has a direct connection to any other player who also selected ROIs in image k (the edge weight is well defined according to Equation 3.14). Thus, since A_k is an normalized strong connected PRG adjacency matrix, which proves A_k is irreducible.

Real elements: With Lemma 3.1 and 3.2, each part of the Equation 3.14 are real number. Thus, of course, the matrix A_k elements are calculated by Equation 3.15 that are real elements.

Non-negative elements: With Lemma 3.1 and 3.2, we have:

$$w_{p,q,k} = \sum_{j=1}^n \sum_{i=1}^m (\text{PRMR}(p, q, i, j, k) (\text{PITC}(p, q, i, j, k) + 2)) \geq \sum_{j=1}^n \sum_{i=1}^m (0 \cdot (-1 + 2)) = 0$$

that $w_{p,q,k}$ has its minimum value when $\text{PRMR}(p, q, i, j, k) = 0$ (for all $i = 1, \dots, m; j = 1, \dots, n$) and $\text{PITC}(p, q, i, j, k) = -1$ (for all $i = 1, \dots, m; j = 1, \dots, n$). Meanwhile,

$$w_{p,q,k} = \sum_{j=1}^n \sum_{i=1}^m (\text{PRMR}(p, q, i, j, k) (\text{PITC}(p, q, i, j, k) + 2)) \leq \sum_{j=1}^n \sum_{i=1}^m (1 \cdot (1 + 2)) = 3mn$$

that $w_{p,q,k}$ has its maximum value when $\text{PRMR}(p, q, i, j, k) = 1$ (for all $i = 1, \dots, m; j = 1, \dots, n$) and $\text{PITC}(p, q, i, j, k) = 1$ (for all $i = 1, \dots, m; j = 1, \dots, n$).

Positive diagonal elements: According to Lemma 3.2, the diagonal elements can be formalized by follows:

$$\begin{aligned} w_{p,p,k} &= \sum_{j=1}^m \sum_{i=1}^m (\text{PRMR}(p, p, i, j, k) (\text{PITC}(p, p, i, j, k) + 2)) \\ &\geq \sum_{j=1}^m \sum_{i=1}^m \left(\frac{|ROI_{p,i,k} \cap ROI_{p,j,k}|}{|ROI_{p,i,k}|} (-1 + 2) \right) \\ &= \sum_{j=1}^m \sum_{i=1}^m \frac{|ROI_{p,i,k} \cap ROI_{p,j,k}|}{|ROI_{p,i,k}|} \\ &= \sum_{i=j}^m \frac{|ROI_{p,i,k} \cap ROI_{p,j,k}|}{|ROI_{p,i,k}|} + \sum_{i \neq j}^m \frac{|ROI_{p,i,k} \cap ROI_{p,j,k}|}{|ROI_{p,i,k}|} \\ &\geq \sum_{i=j}^m \frac{|ROI_{p,i,k} \cap ROI_{p,j,k}|}{|ROI_{p,i,k}|} \\ &= \sum_{i=1}^m \frac{|ROI_{p,i,k} \cap ROI_{p,i,k}|}{|ROI_{p,i,k}|} \\ &= \sum_{i=1}^m \frac{|ROI_{p,i,k}|}{|ROI_{p,i,k}|} \\ &= m > 0 \end{aligned} \tag{3.16}$$

Column stochastic: according to the definition of matrix A , the sum of the column elements are:

$$\sum_q a_{p,q,k} = \sum_q \frac{w_{p,q,k}}{\sum_q w_{p,q,k}} = \frac{\sum_q w_{p,q,k}}{\sum_q w_{p,q,k}} = 1 \tag{3.17}$$

□

From the proof of property of positive diagonal elements, we see that the number “2” is a translation that guarantees $\text{PITC}(p, q, i, j, k)$ lies on $[1, 3]$ which helps us prove this property successfully.

According to **Perron Frobenius theorem** [19, 20] and Theorem 3.3, one can infer that there exists an uniqueness eigenvector $V_k = (TV_{1,k}, \dots, TV_{n,k})$ of A_k , the so-called Perron vector, with an uniqueness eigenvalue $\rho(A_k)$ is the spectral radius of A_k , the so-called Perron root, such that:

$$A_k \cdot V_k = \rho(A_k) \cdot V_k, TV_{i,k} > 0, \sum_{i=1}^n TV_{i,k} = 1.$$

Therefore, we define the trust value of a player:

Definition 3.8. A **Trust Value** $TV_{i,k}$ of player i on image k is a score that equals to the i -th component of the Perron vector of the normalized PRG adjacency matrix A_k .

This definition represents the rating score from player p to player q for a certain image k , as same as the centrality of the player q . With the trust value of players, we propose our classification algorithm:

Algorithm 1: Malicious Player Detection Algorithm

```

input : New Player  $p$ ,
          Trusted Player  $p_1, p_2, \dots, p_m$ ,
          Task Images  $k_1, k_2, \dots, k_{2n}$ ,
          Acceptance Threshold  $\delta$ 
output: Reliability of Player  $p$ 
begin
    counter  $\leftarrow 0$ 
    for  $k \in [k_1, k_2, \dots, k_{2n}]$  do
        if  $k$  is tagged image then
            calculate  $TV_{p,k}, TV_{p_1,k}, \dots, TV_{p_m,k}$ 
            if  $TV_{p,k} \geq \frac{1}{m} \sum_{i=1}^m TV_{p_i,k}$  then
                | counter  $\leftarrow$  counter +1
            end
        end
    end
    if counter  $\geq \delta$  then
        | reliability  $\leftarrow$  true
    else
        | reliability  $\leftarrow$  false
    end
end
```

The purpose of this algorithm is simple, the criterion of classifying new players performs the action that the trust value of a new player should not be less than the mean value of overall trust value of players on image k , which means the tagging performance of new player should not worth than result performance of former players. The acceptance threshold is a customizable parameter that can be set by system manager. For instance, if $\delta = 1$, then the

new player only need pass one singular image of all tagged images; if $\delta = n$ (half images of the task), then the new player have to pass all tagged images;

Note that sometimes new player carries new tags to the system, this will influence the **Tag Vector** calculation, which causes the weight are not computable due to the dimension of the **Tag Vector** of new player and old player inequal. We also address a solution for this issue via the following steps:

- If a new player does not provide new tag: directly perform the calculation with Algorithm 1;
- If a new player carries new tags: Directly drop, it is an unreliable result;
- If a player carries selected tags and also new tags:
 1. Perform the calculation with Algorithm 1 without new tags;
 2. Merge and update all weight vector v via formula 3.14 if the player is reliable,
 3. Otherwise drop and mark the result is unreliable.

3.2.4 DISASTER EVALUATION MODEL

Pooling is an useful technique for training Convolutional Neuron Networks [21, 22], which combine the outputs of neuron clusters at one layer into a single neuron in the next layer. It is a kind of aggregation method for an image region. General pooling techniques are mean pooling, max pooling and stochastic pooling. The he stochastic pooling uses the weighted average value from each of a cluster of neurons at the prior layer [23]. We shift this technique to define our Disaster Evaluation Model.

For a monitor region at time t , we address the **DEM** via disaster level definition as follows:

Definition 3.9. *Let a monitor region is composed by images k_1, \dots, k_n , and players p_1, p_2, \dots, p_m are tagged these images. The player p_j tagged $r_{j,i}$ ROIs in image k_i with $j = 1, \dots, m; i = 1, \dots, n$. The **Disaster Level (DL)** of a monitor region is calculated by the following:*

$$DL = v \cdot \left(\frac{1}{m} \sum_{j=1}^m \sum_{i=1}^n T_{p_j, r_{j,i}, k_i} \right) \quad (3.18)$$

with v is the **System Weight Vector** and $T_{p_j, r_{j,i}, k_i}$ is the **Tag Vector** of the $r_{j,i}$ -th ROI of player p_j in image k_i .

With this Definition 3.9, we can calculate the disaster level for a monitoring region according to Equation 3.18 directly.

3.3 MODEL INITIALIZATION

A cold start of such a system is a common problem in human computation system that is avoided by hiring people to play or learn as long as the number of users or the quantity of data is insufficient. In this system, we have only have to consider one system initialization issue of cold start.

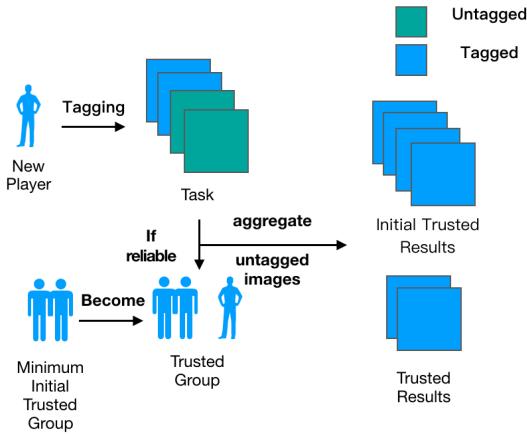


Figure 3.4: Initialization of PRM

The issue appears in the PTG. To initialize the whole system, we need to address an **initial trusted group** for PTG; they shall tagging enough initial trusted result as well as a fixed pre-defined tag list (contains all of the most important tags that organisation need to monitoring) for PTG and then assign the tagged images to new upcoming players. In this system, we do not need to hire people to player and keep the system runs well. This is because: when a new player is reliable, then the result of this player will become reliable. Meanwhile, the trusted group and available dataset grow larger with this step repeatedly, as shown in figure 3.4.

Thus, the final question of our system relate to the minimum number of the initial trusted group. Our PRM is based on graph centrality calculation, which means we need a (at least) two dimensional matrix to perform the overall model calculation. Hence, with the new player, **the minimum number of the initial trusted group is 1**. Then the initial trusted group (one person) with the new player composed to a two dimensional adjacency matrix that makes the model computable. For larger initial trusted groups, the **Trust Value** can be simply initialized to $\frac{1}{n}$ with n is the number of initial trusted group.

4 EVALUATION

This chapter discusses the evaluation criteria with dataset simulation as well as the issues and solutions on social and ethical aspects appears with this system design. Surely, our system

still has limitations, we then discuss the most important two limitations within our system: evaluation outdated and game playability.

4.1 SUCCESS CRITERIAS

4.1.1 MODEL EVALUATION BY SIMULATION

In the previous chapter of design and model, we described the system data flow, which is done by three main step: task generation, malicious player detection and disaster level evaluation as well. The second step malicious player detection is essentially a classification problem that our system detect the new player is reliable or not based on **Trust Value**.

To evaluate our model, a typical classification model performance evaluation metric is to using the Receiver Operating Characteristic (ROC) curve [24], which plots True Positive (on the y-axis) against False Positive and the ideal **surface under the ROC curve** is 1. Nevertheless, before we test the system with real user, one can generate a reasonable random dataset to test the performance of our classification model (**PRM**).

Our player have two different type of inputs: **the ROI, and its Tag Vector**. For a reasonable player data entity, one has to define the ROI selection and its corresponding Tag Vector. To generate reasonable ROI to simulate real user behavior, we discuss the click behavior of a desktop target click behavior first.

The target click behavior has been researched for years and addressed by FFitts Law [25]. It modeled and proved the distribution of click behavior for a certain click goal point is a Gaussian distribution [26]. Thus, with frequency statistic view, the actual ROI(s) for a certain exists, and whatever the user starts from the top right corner or bottom left corner or etc., according to the FFitts Law, the starting click point should follows normal distribution around the actual point, as shown in figure 4.1. Similarly, the landing point of the selection of ROI(s) should also follows a normal distribution.

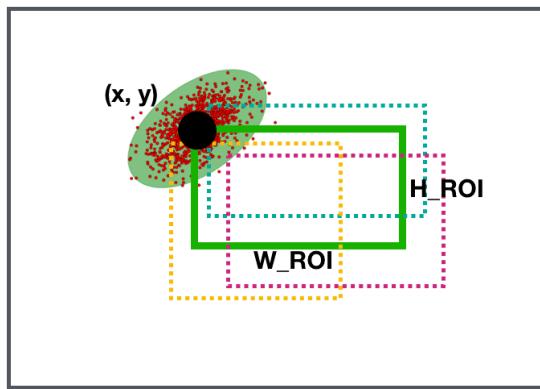


Figure 4.1: Example of ROI Simulation

Therefore, to generate ROI(s), let (x, y) is the player ROI starting point, (H_{ROI}, W_{ROI}) is the height and width pair of this ROI, then we generate noise for the ROI starting points and landing points: $(x + \epsilon, y + \epsilon)$, $(H_{ROI} + \epsilon, W_{ROI} + \epsilon)$ and $\epsilon \sim N(0, \delta)$. For the parameter δ , one can use Maximum Likelihood estimator [27] to perform the inference from all manually ROI selection samples from initial trusted group.

The generation of Tag Vector for a certain image is much simple than ROI's, a randomly pick from initial trusted group is sufficient for the simulation case because these tags are trusted results and a partial randomly selection already introduced the noise in this case.

Eventually, one can perform this random dataset on the system to evaluate **surface under the ROC curve** to evaluate the overall performance of this system (the model shows good performance if the surface approximate to 1), which gives the theoretical evaluation results.

4.1.2 ISSUES ON SOCIAL AND ETHICAL ASPECTS

Intellectual Property Rights: Our disaster monitoring system is an non-commercial project which invites volunteers to contribute to it. It is used to help the UNICEF and other non-profit organizations deliver supplies in disaster areas safer. So the IPR on the annotations should not belong to the volunteer users. In an appropriate manner, user will be informed that his or her contribution to the system is volunteering and the produced data will be used only for monitoring disasters in Syria.

Leakage of data: Malicious issues are not limited to the spite of game playing but also possible on the issue of leakage of our image data. Malicious players also can be terrorists, they can use these satellite image as a free attack resources if we provide the entire region image to the game player, which leads to the data insecurity. Therefore, we cut big satellite image into small segmentations in our system to prevent data leakage to ordinary players for the data security reason.

Information Loss We cut big region images into small fragement areas to prevent leakage of data. However this method will cause some information loss problem if some important ROIs are located at the intersection of two dividing lines. A possible solution for this limitation is to consider “Half Shifting” cut. The “Half Shifting” cut have two grids, the one obtained from the other shift the original grid to the right and to the bottom by half the breath respectively hight of a ROI, as shown in figure 4.2. Note that this solution that only increases the number of region pictures, it does not influence any model and system design.

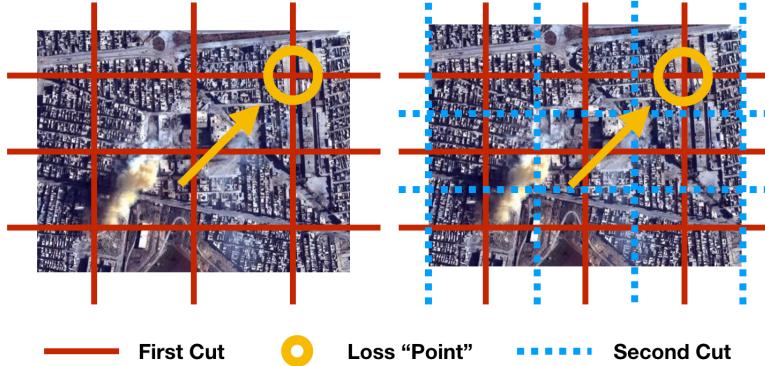


Figure 4.2: **Information Loss:** Information loss may occurs on the intersection lines, as shown on the left; a possible solution is to perform a “Half Shifting” cut

4.2 LIMITATIONS OF THE SYSTEM

4.2.1 EVALUATION OUTDATED

A limitation occurs in a situation that our PRG network is based on image dependent prespective, that leads, each calculated **Disaster Level** get invalid if the region image is outdated. We assume the satellite machine monitors an region and take pictures between intervals. However, our evaluation the model only calculate the disaster level at a unique moment, which means the disaster level need transvaluation when a new image comes out. If none of all the new images from a region get evaluate, then the disaster level will not be calculated.

Note that the disaster level of a disaster region over time is essentially a non-stationary process [28] due to the paroxysmal disaster situation. One can directly perform time series prediction method [29] on the disaster level time series. Unfortunately, there is no space to cover them in depth.

4.2.2 GAMEPLAY AND PLAYABILITY

The GWAP collects satellite photos of disaster areas. But even if in the disaster areas, not every part of the areas has a disaster. Most parts of the earth are lake, forest, desert and so on, which means the users may meet the situation that there is no available ROI in several continuous rounds. Obviously, it will decrease the playability and enjoyment of the game. Our system is just a very simple tagging game at present, users can not get enough enjoyment they want in it. And it is too reliant on the unpaid volunteers to donate their time to contribute information. We should make the system more interesting and appealing in the future work.

5 CONCLUSIONS AND FUTURE WORKS

5.1 CONCLUSIONS

In this report, we proposed a comprehensive design of a human computation system for disaster monitoring, and we discussed its mathematical foundations as well as the possible issues caused by this system, then gives few options to solve these issues.

In the chapter of **Functionalities**, we illustrated a mockup of GWAPs-based disaster monitoring human computation system for game players as well as stakeholders, and then described its necessary functions and interaction logic. On thoughts of system implementation, we discussed the possible technology stack of this system on the Web.

Afterward, in the chapter of **Design and Models**, we modeled the entire system theoretically in details that make sure it can run consistently. First of all, we defined a **Player Rating Model** based on eigenvalue centralities for calculating trust value of a player, and then we put forward an algorithm that can be used in malicious user detection. As justification, we proved the correctness of this model. Meanwhile, as the data aggregation, we transferred the problem of calculating disaster level of regions into processing the expectancy of user tagging task inputs and proposed the **Disaster Evaluation Model**. Surely, we addressed the solution of cold start of the human computation system. It is worth mentioning that the minimum initial trusted group under this scheme design only requires one person theoretically.

Furthermore, in the chapter of **Evaluation**, we discussed theoretical evaluation criteria for this system, and then declared the challenges and corresponding solutions for facing issues like data security, information leakage and loss, malicious detection as well as the lack of players. Undoubtedly, the current system design still contains defects. Thus, we presented three analysis and possible improvements for evaluation outdated and gameplay playability.

5.2 FUTURE WORKS

Our system was described in general. We collect human inputs by ROI tagging tasks, which means any other human computation system that related to ROI tagging tasks can easily use this system backend design. In addition, due to the fact that we do not have enough user inputs at present, we use a certainty algorithm instead of uncertainty probabilistic-based algorithm to detect malicious groups. Considering malicious detection is classification problem, which separate users into trusted groups and untrusted groups. One can apply any classification machine learning algorithms that are more suitable for the detection of malicious groups if our user input dataset is large enough.

Besides, as we mentioned before, the game player may encounter a situation that there is no ROI in some pictures which contain only landscapes like mountains, rivers and forests. In this case, the game playability is significantly reduced. However, for future work, we can filter out those images from our image database previously with collaborative computing of image

recognition technique, so that one can collect more data from the game and make our image tagging game more efficiently.

ACKNOWLEDGEMENTS

The authors would like to thank Prof. François Bry first for his great and important suggestions on the model statements as well as information leak and loss problems of the disaster monitoring system; we also thank Yingding Wang for his helpful discussions on system functionalities design and the model rationalizations as well as evaluations; Finally, we also thank our schoolmate Huimin An for his inspiration of Bayesian perspective that helps us handling human inputs with new tags successfully.

The resources of this project are open source on GitHub, such as paper L^AT_EX code, mockup drafts as well as lab session beamer slides:

<https://github.com/changkun/hc-ss17-disaster-monitoring>.

REFERENCES

- [1] Unicef. *The state of the world's children*. 1998. Unicef, 1994.
- [2] Unicef. UNICEF - Humanitarian Action for Syrian Refugees - Situation Reports. https://www.unicef.org/appeals/syrianrefugees_sitreps.html, 2017. [Online; accessed 31-July-2017].
- [3] Jiqun Zhang, Chenghu Zhou, Kaiqin Xu, and Masataka Watanabe. Flood disaster monitoring and evaluation in China. *Global Environmental Change Part B: Environmental Hazards*, 4(2):33–43, 2002.
- [4] Alexander J Quinn and Benjamin B Bederson. Human computation: a survey and taxonomy of a growing field. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1403–1412. ACM, 2011.
- [5] Alexander J Quinn and Benjamin B Bederson. A taxonomy of distributed human computation. *Human-Computer Interaction Lab Tech Report, University of Maryland*, 2009.
- [6] Luis Von Ahn and Laura Dabbish. ESP: Labeling Images with a Computer Game. In *AAAI spring symposium: Knowledge collection from volunteer contributors*, volume 2, 2005.
- [7] Luis Von Ahn. Games with a purpose. *Computer*, 39(6):92–94, 2006.
- [8] Luis Von Ahn and Laura Dabbish. Designing games with a purpose. *Communications of the ACM*, 51(8):58–67, 2008.
- [9] Christoph Wieser, François Bry, Alexandre Bérard, and Richard Lagrange. ARTigo: build-

- ing an artwork search engine with games and higher-order latent semantic analysis. In *First AAAI Conference on Human Computation and Crowdsourcing*, 2013.
- [10] Emma Graham-Harrison. 'I couldn't take anything except dignity': stories of the leaving of Aleppo. <https://www.theguardian.com/world/2016/dec/23/i-couldnt-take-anything-except-dignity-people-aleppo-syria-on-fleeing-city>, 2016. [Online; accessed 31-July-2017].
 - [11] Luis Von Ahn and Laura Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM, 2004.
 - [12] James Wu and Stephen Coggeshall. *Foundations of predictive analytics*. CRC Press, 2012.
 - [13] Huan Liu, Farhad Hussain, Chew Lim Tan, and Manoranjan Dash. Discretization: An enabling technique. *Data mining and knowledge discovery*, 6(4):393–423, 2002.
 - [14] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.
 - [15] Phillip Bonacich and Paulette Lloyd. Eigenvector-like measures of centrality for asymmetric relations. *Social networks*, 23(3):191–201, 2001.
 - [16] Raimundo Real and Juan M Vargas. The probabilistic basis of Jaccard's index of similarity. *Systematic biology*, 45(3):380–385, 1996.
 - [17] Paul Jaccard. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bull Soc Vaudoise Sci Nat*, 37:547–579, 1901.
 - [18] Karl Pearson. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58:240–242, 1895.
 - [19] Oskar Perron. Zur theorie der matrices. *Mathematische Annalen*, 64(2):248–263, 1907.
 - [20] Ferdinand Georg Frobenius. Über Matrizen aus nicht negativen Elementen. 1912.
 - [21] Dan C Ciresan, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and Jürgen Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1237. Barcelona, Spain, 2011.
 - [22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
 - [23] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012.
 - [24] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36, 1982.

- [25] Xiaojun Bi, Yang Li, and Shumin Zhai. FFitts law: modeling finger touch with fitts' law. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1363–1372. ACM, 2013.
- [26] NR Goodman. Statistical analysis based on a certain multivariate complex Gaussian distribution (an introduction). *The Annals of mathematical statistics*, 34(1):152–177, 1963.
- [27] Søren Johansen and Katarina Juselius. Maximum likelihood estimation and inference on cointegrationwith applications to the demand for money. *Oxford Bulletin of Economics and statistics*, 52(2):169–210, 1990.
- [28] Peter J Brockwell and Richard A Davis. *Time series: theory and methods*. Springer Science & Business Media, 2013.
- [29] Vitaly Kuznetsov and Mehryar Mohri. Learning theory and algorithms for forecasting non-stationary time series. In *Advances in neural information processing systems*, pages 541–549, 2015.