Final Team Report for HC System: A Novel GWAPs Disaster Monitoring System

Team: Hotpot
Changkun Ou : <11406972>
Yifei Zhan : <Matrikelnummer>
Zhe Li : <11194275>

July 14, 2017

CONTENTS

ABSTRACT Abstract test

1 Introduction

1.1 Related information on the topic field

1.2 Purpose of HC system

1.3 Human contribution to the system

2 Functionality of a novel HC system

2.1 Functionality as seen by a user

2.2 Functionality as seen by a stakeholder

2.3 Incentivization concept

3 System Design

In this section we describes the overall design of our disaster monitoring backend system design.

3.1 SYSTEM ARCHITECTURES

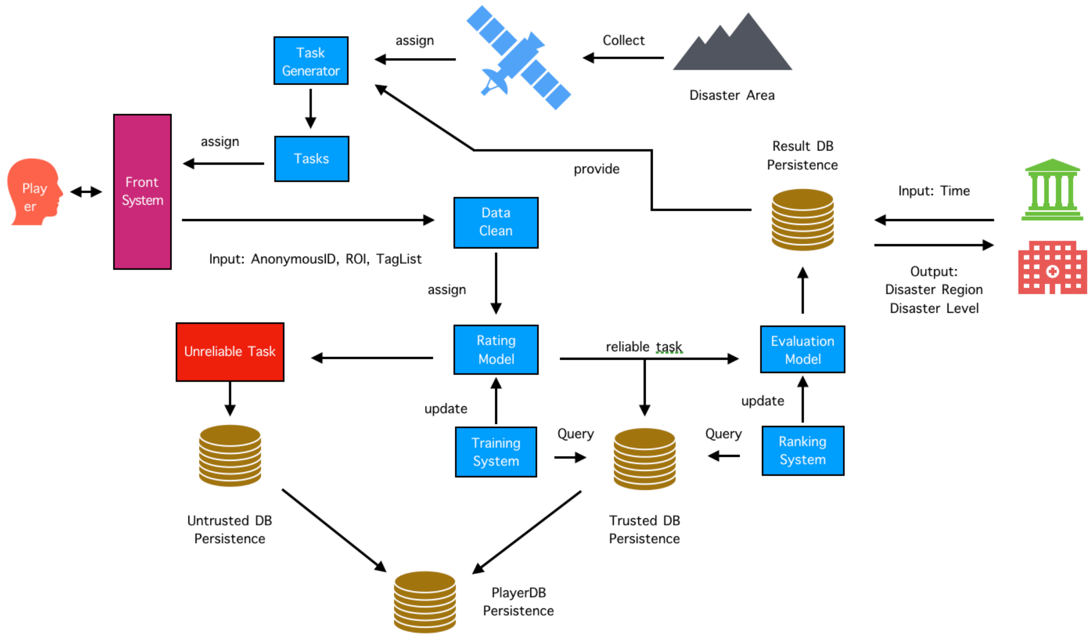

Figure 3.1: System Design Overview

The system contains two different type of databases. The first databases **PlayerDB** combines with **TrustedDB** and **UntrustedDB** where presistent the player inputs whether the overall result is reliable or not. We designed a task generator that combines trusted results and seperate new satellite area images assign to upcomming players. A reliable player shall pass the system **Player Rating Model**. Once the task result from new player is reliable, then the system will reuse the player input into our **Disaster Evalutation Model** and presistent it in the second database **ResultDB**. Stakeholder make querys to this monitoring database. Figure 3.1 illustrate the overall disaster system design.

3.2 SYSTEM COMPONENTS

3.2.1 DATABASE FIELDS

For the convinience of model establishment, we describe the system database PlayerDB fields as well as the fields of database ResultDB in the follows listing 1 and 2:

```
[
 {
  "anonymous_id": number,
  "reliable": boolean,
  "trust_value": number
  "tasks": [
   {
    "image": image_path,
    "at": time,
    "ROI": [
     {
      "latitude": number,
      "longitude": number,
      "tags": [tag1, tag2,
   ...]
     }, ...
    ]
   }
  ]
 }, ...
]
```

Listing 1: Player Database Fields

```
[
 {
  "area_id": number,
  "disaster_level": number,
  "history": [
   {
    "at": time,
    "image": image_path,
    "ROI": [
     {
      "latitude": number,
      "longitude": number,
      "tags": [tag1, tag2,
   ...]
    }
   ]
  },
  ...
 ]
}, ...
]
```

Listing 2: Results Database Fields

In this disaster monitoring system, our participant do not need to register an account, and the system shall assign a anonymous_id for each player, this function significantly accelerate player to particpate to this game. Thus, the PlayerDB stores the anonymous_id to detect the same players if they participate next time. The player will accomplish different game tasks, each task result shall stores in the tasks filed.

In the ResultDB, an area ID is unique, and assigned by our system, the disaster_level field represents the level of this area. Each area shall be evaluated by our player, and the evaluation history stores in history field.

To explain other fileds, we describe few basic definition for the system model.

Definition 3.1. The **Region of Interests (ROI)** ROI_i is a player selected area of player i.

Definition 3.2. The **tags vector** T_i of player i is indicated by a vector where the components represent by the count of all tags:

$$T_i = (|\mathrm{tag}_1|, |\mathrm{tag}_2|, ..., |\mathrm{tag}_n|)$$

where

- n is the number of current exist tags;
- $|\mathrm{tag}_n|$ is the occurrance of tag_n.

For instance, there are 5 different tags $\mathrm{tag}_1, \mathrm{tag}_2, \mathrm{tag}_3, \mathrm{tag}_4, \mathrm{tag}_5$ exist in the current system, player i generates tags list $\{\text{tag}_1, \text{tag}_2, \text{tag}_3\}$, player j generates tag list $\{\text{tag}_4, \text{tag}_4, \text{tag}_5\}$. Then T_i of player i is $(1,1,1,0,0)$ and T_j of player j is $(0,0,0,2,5)$.

Definition 3.3. The **weight vector** $v = (p(\text{tag}_1), p(\text{tag}_2), ..., p(\text{tag}_n))$ **of all tags** can be calculated by the following equation 3.1:

$$p(\text{tag}_i) = \frac{|\text{tag}_i|}{\sum_{j=1}^{n} |\text{tag}_j|} \tag{3.1}$$

where

- n is the number of current exist tags;
- $|\text{tag}_i|$ is the occurrance of tag_i.

3.2.2 Player Task Generator

The **Player Task Generator (PTG)** combines images from satellite and ResultDB. In the first step, as we discussed before, to solve the imformation leakage problem, PTG shall split a monitoring region into $m \times n$ small pieces of images, and also assign a unique **areaID** for each pieces, i.e. $(\text{areaID}, \text{time})$ specifice a unique image for user tasks.

The second generate step is to retrieve tagged images from **ResultDB**. Then combine all images as a user task assign to a new upcomming player. Each user task contains half of untagged images and half of tagged images.

In short, The Data Model (only ouput here) for PTG is: $\{(\text{areaID}_1, \text{time}_1), ..., (\text{areaID}_n, \text{time}_n)\}$ with areaID_1 to $\text{areaID}_{\lfloor n \rfloor}$ are from satellite and $\text{areaID}_{\lfloor n \rfloor + 1}$ to areaID_n are from **ResultDB**.

3.2.3 Player Rating Model

This subsection describes the Player Rating Model inside our Disaster Monitroing system. PageRank was first proposed by Lary Page [1] and applied to social analysis in [2]. 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 and use eigenvalue as a **Trust Value (TV)** for each players to distinguish manicious players.

Considering a partial fully connected directed graph betwen players. Each player is a node of the Player Rating Graph (PRG) as illustrate in figure 3.2.

To define the edge weight, according to the database feild design of a player, each player output ROIs for each task region of a player task, and each ROI contains a tags list, thus, one can use three festures: ROI, tags, TV.

Definition 3.4. The weight from player i to player j can be formalized as follows formula 3.2:

$$w_{ij} = \sum_{\text{ROI} \in \text{ROIs}} \text{TV}_i \times \frac{\text{ROI}_i \cap \text{ROI}_j}{\text{ROI}_i} \left(2 - \frac{Cov(T_i, T_j; v)}{Cov(T_i, T_i; v) \times Cov(T_j, T_j; v)} \right) \tag{3.2}$$

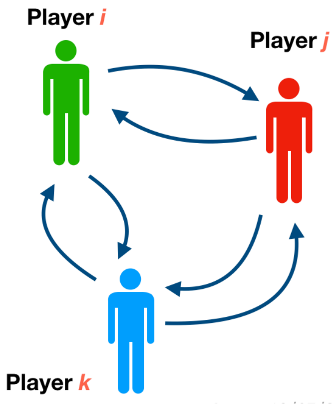

Figure 3.2: Player Rating Model

where

- TV_i is the trust value of player i;
- ROI_i is the selected ROI from player i;
- T_i is the tags vector of player i;
- $Cov(x, y; v)$ is the weighted covariance of x and y via v;
- v is the weight vector of all tags.

The first part of the definition $\sum_{\text{ROI}\in\text{ROIs}}$ summarized all possible ROI between player i and player j. The theioretically item of this formular is the number of ROI from player i multiply the number of ROI from player j. Nevertheless, it can be significantlly decreased in this particular scienario. Considering player i and player j with two ROIs as illustrate in figure 3.3.

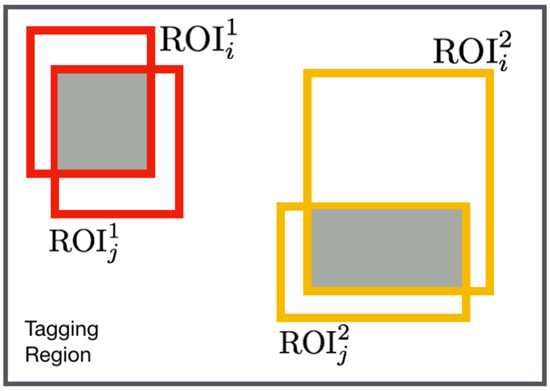

Figure 3.3: Two players with two ROIs

One can expand equation 3.2 as follows formula 3.3:

$$w_{ij} = \mathrm{TV}_i \times \left(2 - \frac{Cov(T_i,T_j;v)}{Cov(T_i,T_i;v)\times Cov(T_j,T_j;v)}\right)\times$$
$$\left(\frac{\mathrm{ROI}_i^1\cap \mathrm{ROI}_j^1}{\mathrm{ROI}_i^1} + \frac{\mathrm{ROI}_i^1\cap \mathrm{ROI}_j^2}{\mathrm{ROI}_i^1} + \frac{\mathrm{ROI}_i^2\cap \mathrm{ROI}_j^1}{\mathrm{ROI}_i^2} + \frac{\mathrm{ROI}_i^2\cap \mathrm{ROI}_j^1}{\mathrm{ROI}_i^2}\right) \quad (3.3)$$

Fortunately, the second and the third part of the expation are equal to zero.

We call the second part $\mathrm{TV}_i \times \frac{\mathrm{ROI}_i \cap \mathrm{ROI}_j}{\mathrm{ROI}_i}$ of formula 3.2 as **Matching Area Retio (MAR)**. It was inspired by a common computer vision criteria, the so called Intersection over Union (IoU), also called Jaccard Index in mathematics[3], which is the standard performance measure that is commonly used for the object category segmentation problem. Nevertheless, MAR is not equal to the IoU of ROIs of player i and player j since it only use the ROI of player i as denominator instead of the union of ROIs of player i and player j, which leads the differnce between MAR and IoU. There are two reason to use MAR instead of IoU: Firstly, IoU as weight of graph causes the directed graph to an undirected graph due to the IoU of player i to j is as same as the IoU of player j to i; Furthermore, player i as the evaluator from i to j should be the performance base.

The third part $\frac{Cov(T_i,T_j;v)}{Cov(T_i,T_i;v)\times Cov(T_j,T_j;v)}$ of formula 3.2 is applied by Weighted Pearson Correlation Coefficient.

To calculate the eigenvalue of the ajacency matrix of PRG, one can use the normalized ajacency matrix through the following formula 3.4:

$$A = (a_{ij}) = (\frac{w_{ij}}{\sum_j w_{ij}}) \quad (3.4)$$

Theorem 1. *Matrix A is irreducible, real, non-negative, column-stochastic, and diagonal element being positive.*

Proof. **Irreducibility**: A is normalized through an ajacency matrix of a strong connected player rating graph, which proves A is irreducible.

Real elements: Trivial.

Non-negative elements: We only need to prove TV_i, $\frac{\mathrm{ROI}_i \cap \mathrm{ROI}_j}{\mathrm{ROI}_i}$ and $2 - \frac{Cov(T_i,T_j;v)}{Cov(T_i,T_i;v)\times Cov(T_j,T_j;v)}$ are non-negative respectively. TV_i is the eigenvalues of normalized graph ajacency matrix, thus the codomain of TV_i lies $(0,1]$; For MAR, its range is obviously from 0 to 1, which lies $[0,1]$; For $2 - \frac{Cov(T_i,T_j;v)}{Cov(T_i,T_i;v)\times Cov(T_j,T_j;v)}$, the Pearson Correlation lies on $[-1,1]$, then this part lies on $[1,3]$. Three parts are non-negative.

Positive diagonal elements: The diagonal elements can be formalized by follows:

$$w_{ii} = \sum_{\mathrm{ROI}\in \mathrm{ROIs}} \mathrm{TV}_i \times \frac{\mathrm{ROI}_i \cap \mathrm{ROI}_i}{\mathrm{ROI}_i}\left(2 - \frac{Cov(T_i,T_i;v)}{Cov(T_i,T_i;v)\times Cov(T_i,T_i;v)}\right) = \sum_{\mathrm{ROI}\in \mathrm{ROIs}} \mathrm{TV}_i > 0$$

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

$$\sum_i \frac{w_{ij}}{\sum_j w_{ij}} = \frac{\sum_i w_{ij}}{\sum_j w_{ij}} = 1$$

□

We has proved the existence and uniqueness of eigenvalues of normalized PRG ajacency matrix, one can use the corresponding eigenvalues to represent the trust value of players. Thus, we have:

Definition 3.5. A **Trust Value** TV_i of player i represents by the i-th eigenvalue of normalized PRG ajacency matrix A

This definition can represents the rating score from i to j. With the trust value of players, we propose our classification algorithm:

Algorithm 1: Player Classification Algorithm

---

input : anonymous IDs, TVs
output: (anonymous_id, isReliable)
Calculate TV_{new} as the trust value of player new ;
if $TV_{new} \geq \frac{1}{|players|} \sum_{i \in players} TV_i$ **then**
  return (anonymous_id, true)
else
  return (anonymous_id, false)
end

In this algorithm, the criterion of classify new players performs the action that the trust value of new player should not less than the mean value of overall trust value of players, which means the tagging performance of new player should not worth than result performance of former players.

Terefore in short, the input and output Data Model of PRM are as follows. For input: (anonymous_id, area_id, time, ROIs, tags); For model output: (anonymous_id, TV).

3.2.4 Disaster Evaluation Model

For an area at time t, we address the **Disaster Evaluation Model (DEM)** via disaster level definition as follows:

Definition 3.6. The **Disaster Level (DL)** of a monitor region is calculated by each area components:

$$DL = \sum_{\text{area} \in \text{region}} DL_{\text{area}}$$

where DL_{area} is calculated by its corresponding tag vector:

$$DL_{\text{area}} = \sum_{i=1}^{n} v_i \times |\text{tag}_i|$$

with n is the number of current exist tags, and $|\text{tag}_i|$ is the occurrance of tag_i in the corresponding area.

System like ESP[4], ARTigo[5] has proved that human inputs are valuable and useful.

Note that sometimes player carries new tags for our system, we also address a solution for this issue via the following steps:

- When a player carries predefined tags: Trivial;
- When a player carries new tags: Directly drop, it is an unreliable result;
- When a player carries predefined tags and also new tags: calculate the trust value without new tags; merge and update all weight vector v via formula 3.2 if the player is reliable, otherwise drop and mark the result is unreliable.

With this definition 3.6, we can calculate the disaster level for a monitoring region. To sum up, the input and the output Data Model of DEM addresse as follows. For input: (time), (area_id) or (area_id, time); For output: (area_id, time, disaster_level).

3.2.5 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 our system, we have two different cold start problem.

The first cold start problem appears in the PTG. To initialize the whole system, we need address a initial trusted group for PTG, they shall tagging ennough initial trusted result for PTG and then assign to new upcomming players. When a new player is reliable, then the result of this player will become reliable. Meanwhile, the trusted group and avaliable dataset become larger with this step repeatedly, as shown in figure 3.4.

The second cold start problem appears in PRM. According to the definition **??** of PRG, the weight of PRG was defined by the trust value of all players. Nevertheless the initial trusted group has no trust value. Thus we need a initial value for TV. Note that TV_i is in between of 0 and 1, thus:

$$TV_i^{\text{init}} = \frac{1}{|\text{players}^{\text{init}}|}$$

with $|\text{players}^{init}|$ is the number of initial trusted group.

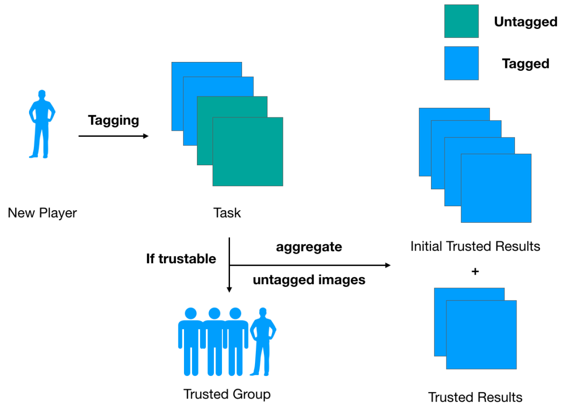

Figure 3.4: Cold Start of PTG

SUMMARY We have described our monitoring system in general and also detailed. It contains the following components:

- Presistent Databases
- Player Task Generator
- Player Rating Model
- Disaster Evaluation Model

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

4 System Evaluation and Success Criteria

4.1 Evaluation and success criteria

4.1.1 Model Evaluation

Malicious player detection is a classification problem. One can generate random data and test the Rating Model through accuracy and recall, even ROC curve [6].

The click behavior has been researched for years and address by FFitts Law [7]. It modeled and proved the distribution of click behavior for a certain click goal point is a normal distribution. Thus, with probablistic view, the top left corner of ROI exists, then the user click selection for this point should follows normal distribution, as shown in figure 4.1.

Therefore, to generate ROIs, let (x, y) is the player ROI start point, (H_{ROI}, W_{ROI}) is the height

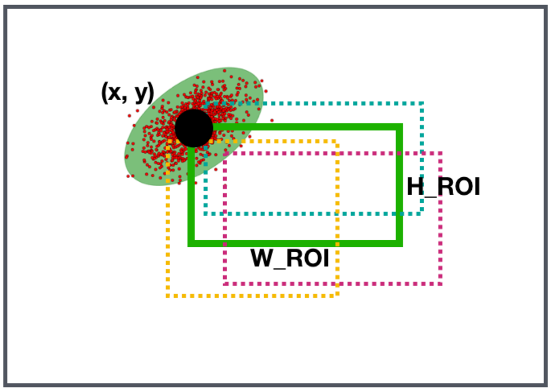

Figure 4.1: Data Simulation

and width pair of this ROI, then we generate the random dataset for these variables by a given parameter δ: $(x, y) \sim (x + N(0,\delta), y + N(0,\delta)), (H_{ROI}, W_{ROI}) \sim (H_{ROI} + N(0,\delta), W_{ROI} + N(0,\delta))$. To generate tags, we propose randomly pick random number of tags.

Then once can perform this random dataset on our system to evaluate the classification accuracy and recall rate to evaluate the overall performance of this system.

4.1.2 Issues on Social and Ethical Aspects

4.2 Limitation of the System

4.2.1 Evaluation Outdate

A limitation occurs in our social network based model is each disaster level evaluation get invalid if the region image outdate. We assume the satellite monitors a region and take picutre between intervals. However, our evaluation model only calculate the disaster level in a unique moment, which means the disaster level need transvaluation when a new image come out. If our player are not enough so that the region images always have to wait new evaluation, then the disaster level will never be calculated.

A possible solution is to consider the region disaster level history as a time series. Then we can apply some prediction method for it. For instance, we have time series: $(t_1, t_2, t_3, ..., t_n)$ and its corresponding disaster level: $(DL_1, DL_2, DL_3, ..., DL_n)$. Then we can use these time series to predict the disaster level at time $t_(n + 1)$.

At the same time, we also have the historical data of trust value of a player. We can also use time series prediction to predict the players trust value. But in all of these, the time series of disaster level is not stationary but the time series of trust value is stationary.

5 Conclution & Future Works

5.1 Conclution

5.2 Possible extensions of the HC system

5.3 Thoughts on interaction with other HC system

6 Acknowledgements

The authors would like to thanks Prof. Bry, Yingding Wang, Huiming An for their useful suggestions and discussion.

References

[1] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web.," tech. rep., Stanford InfoLab, 1999.

[2] P. Bonacich and P. Lloyd, "Eigenvector-like measures of centrality for asymmetric relations," *Social networks*, vol. 23, no. 3, pp. 191–201, 2001.

[3] R. Real and J. M. Vargas, "The probabilistic basis of jaccard's index of similarity," *Systematic biology*, vol. 45, no. 3, pp. 380–385, 1996.

[4] L. Von Ahn and L. Dabbish, "Labeling images with a computer game," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 319–326, ACM, 2004.

[5] C. Wieser, F. Bry, A. Bérard, and R. Lagrange, "Artigo: building an artwork search engine with games and higher-order latent semantic analysis," in *First AAAI Conference on Human Computation and Crowdsourcing*, 2013.

[6] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve.," *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.

[7] X. Bi, Y. Li, and S. Zhai, "Ffitts law: modeling finger touch with fitts' law," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1363–1372, ACM, 2013.