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

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

July 3, 2017

CONTENTS

ABSTRACT Abstract test

1 INTRODUCTION

Introduction cite test [4]
The contribution of this report are as follows:

- a novel Human Computation system based on Game with Purpose for Disaster Monitoring;
- xxx

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

A player can finish infinity Round tasks, a Round task contains N tagging tasks, the player tagging task is to:

- Select a Region Of Interests(ROI) upon the presented satellite image;
- Tag the ROI from a provided tag list or input their own tag, the provided tag list contains: $T_1, T_2, , T_n$, other(input needed)

Note that:

- A ROI is a sub-rectangle-window of a image;
- Multiple selections;
- Anyone can directly participant without registration, but system records an ID

2.2 FUNCTIONALITY AS SEEN BY A STAKEHOLDER

2.3 INCENTIVIZATION CONCEPT

2.3.1 TASK GENERATOR

A task generator combines images from satellite and Result DB:

- Split a certain monitoring area image to pieces of images;
- Mix images from Result DB and pack as a Tagging Task which to be assigned to player.

2.3.2 PLAYER RATING MODEL

Players input vector:

(anonymous_id, image, event_time, ROI, tag_list)

Model output:

(anonymous_id, trust_value)

Note that:

- $(anonymous\_id, image, event\_time, ROI)$ is the primary key of the input vector;
- A player can generate multiple vectors to rating system even for same image;
- The event_time is the capture time of the satellite image.

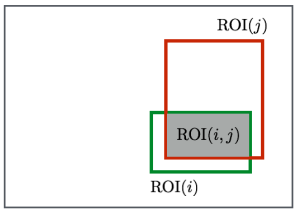

Figure 2.1: Weight Definition Visualization

For a certain image img at time t, Rating: player $i \rightarrow$ player j:

$$w_{ij} = \sum_{\text{ROI} \in \text{ROIs}} \frac{\text{ROI}(i,j)}{\text{ROI}(i)} \times \frac{Cov(\text{tags}(i), \text{tags}(j))}{\text{var}(\text{tags}(i))\text{var}(\text{tags}(j))} \geq 0$$

Normalized Adjacency Matrix:

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

Obviously, A is **irreducible, real, non-negative, column-stochastic, and diagonal element being positive,** then eigenvalue of A is the player trust value.
When a new player tagging task need to be rated,

- which means we need introduce a new node to the graph
- need calculate the trust value of new graph
- let t is the trust value of new player
- if t >= mean(old_eigenvalues), then it is a reliable player, otherwise drop it.

2.3.3 Disaster Level Evaluation Model

Query input:

$$(\text{time}) or (\text{area\_id}) / (\text{area\_id}, \text{time})$$

Model output:

$$(\text{area\_id}, \text{time}, \text{disaster\_level})$$

Note that:

- All results are evaluated from reliable tasks
- Evaluation Model generated by all reliable history

Now we have trusted results, each area has its tagging history.
For an area at time t, define disaster level as follows:

$$v_{area} = \frac{\sum_{\text{tag}\in\text{tags}} w_{tag} \times \#(\text{tag})}{\sum_{area\in areas}\sum_{\text{tag}\in\text{tags}} w_{tag} \times \#(\text{tag})}$$

where w_{tag} is pre-defined weight by system, $\#(tag)$ is the occur number of a tag.
Return value:

- disaster region: $\cup_{ROI\in ROIs} ROI$
- disaster level: v_{area}

2.3.4 Data Persistence

Trusted DB Fields:

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

```

Listing 1: Trusted Database Field

Result DB Fields:

```
[
    {
        area_id": number,
        "history": [
            {
                at_time: time,
                "image": image_path,
                "ROI": [
```

```
                        {
                            "latitude": number,
                            longitude": number,
                            "tags": [tag1, tag2, ...]
                        }
                    ],
                    "disaster_level": number
                }
            ]
        }
    ]

```

Listing 2: Results Database Field

3 System Design and UI Elements

3.1 System Architectures

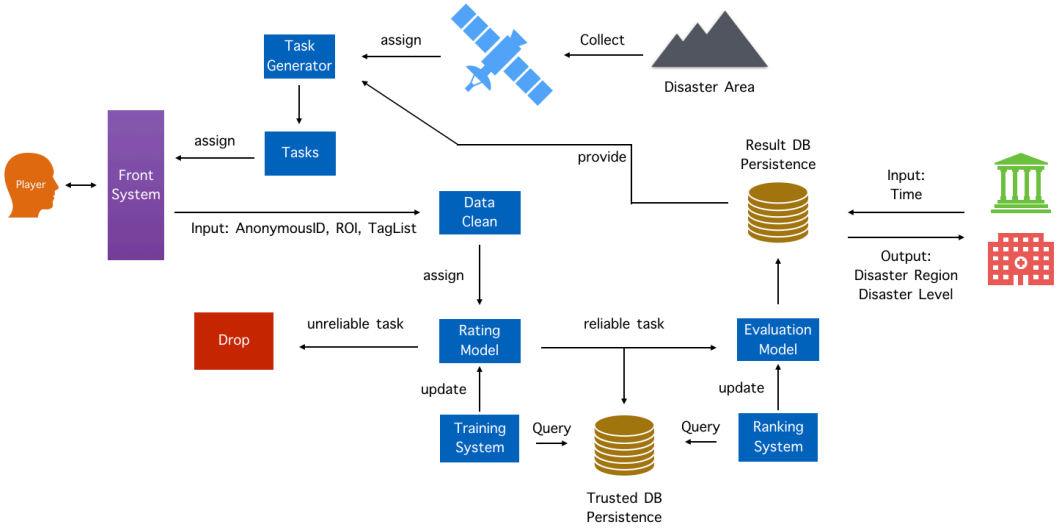

Figure 3.1: System Design Overview

3.2 Algorithm for Data Aggregation

3.3 Technologies used for implementation

For a prototype, we decided to use the following framework to implement everthing:

- Polymer
- Node.js
- MongoDB

3.4 Front End

3.5 Back End

3.6 User Interfaces of the system

3.7 Summary

- Task Generator combines trusted results assign to players;
- Always treat player as new player, but integrated as old player if exists;

- Use ROI matching rate as graph edge weight, eigenvalue as trust value of player;
- Disaster Evaluation use pre-defined weight, then defined the disaster level

4 System Evaluation and Success Criteria

4.1 Limitation of the System

4.2 Evaluation and success criteria

4.2.1 Model Evaluation

4.2.2 Issues on Social Aspects

4.2.3 Issues on Ethical Aspects

5 Future Works

In this report, we present a disaster monitoring system, which aggregate human tagging input based on Network Analysis.

5.1 Possible extensions of the HC system

5.2 Thoughts on interaction with other HC system

REFERENCES

[1] Shivani Agarwal. Ranking Methods in Machine Learning A Tutorial Introduction. *SIAM International Conference on Data Mining*, 2010.

[2] Phillip Bonacich and Paulette Lloyd. Eigenvector-like measures of centrality for asymmetric relations. *Social Networks*, 23(3):191–201, 2001.

[3] François Bry. Human Computation and Economics. (March):1–12, 2012.

[4] François Bry. Human Computation-Enabled Network Analysis for a Systemic Credit Risk Rating. *Handbook of Human Computation*, pages 1–31, 2013.

[5] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. *Proceedings of the 22nd international conference on Machine learning - ICML '05*, pages 89–96, 2005.

[6] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to rank. *Proceedings of the 24th international conference on Machine learning - ICML '07*, pages 129–136, 2007.

[7] European Commission. Syria crisis. *Forced Migration Review*, (March):3, 2016.

[8] Kevin Fran, Marco Saerens, and Pierre Dupont. Classification in Graphs using Discriminative Random Walks.

[9] Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. The Eigentrust algorithm for reputation management in P2P networks. *Proceedings of the twelfth international conference on World Wide Web - WWW '03*, page 640, 2003.

[10] Edith Law and Luis Von Ahn. Input-Agreement : A New Mechanism for Collecting Data Using Human Computation Games. *Proc. SIGCHI Conference on Human Factors in Computing Systems*, pages 1–10, 2009.

[11] Brian Mcfee and Gert Lanckriet. Metric Learning to Rank. *Icml*, pages 775–782, 2010.

[12] Yukikazu NAKAMOTO. A Short Introduction to Learning to Rank. *IEICE Transactions on Information and Systems*, E94-D(1):1–2, 2011.

[13] Luis von Ahn and Laura Dabbish. Designing games with a purpose. *Communications of the ACM*, 51(8):57, 2008.

[14] LI Yong-Jun and DAI Ya-Fei. Research on Trust Mechanism for Peer-to-Peer Network. *Chinese Journal of Computers*, 33(3):390–405, 2010.