Quality Assurance for Human Computation Based Recommendation

Master Defense Presentation

Michael Schwarz
June 13, 2017

Graz University of Technology
Introduction
Motivation

• Recommender systems are ubiquitous
Motivation

• **Recommender systems** are ubiquitous
• These systems are usually based on knowledge
Motivation

- **Recommender systems** are ubiquitous
- These systems are usually based on **knowledge**
- Reliable but **expensive** if entered by small number of **experts**
Motivation

- **Recommender systems** are ubiquitous
- These systems are usually based on **knowledge**
- Reliable but **expensive** if entered by small number of **experts**
- **Unreliable** but cheap if entered by regular **users**
Motivation

- **Recommender systems** are ubiquitous
- These systems are usually based on **knowledge**
- Reliable but **expensive** if entered by small number of **experts**
- **Unreliable** but cheap if entered by regular **users**
- **Combine approaches** to reliably and cheaply collect knowledge
• **Design and implement** a web-based generic recommender platform
Overview

- Design and implement a web-based generic recommender platform
- Add mechanisms to collect data from regular users
Overview

- **Design and implement** a web-based generic recommender platform
- Add mechanisms to **collect data** from regular users
- Develop techniques to ensure the **quality** of the collected data
Overview

• Design and implement a web-based generic recommender platform
• Add mechanisms to collect data from regular users
• Develop techniques to ensure the quality of the collected data
• Efficiently distribute tasks to users to improve the knowledge base
Recommender Systems
Recommender Systems

• There are different types of recommender systems
Recommender Systems

- There are different types of recommender systems
- They all recommend products/items...
There are different types of recommender systems. They all recommend products/items... ...but use different techniques to find the best item(s).
Recommender Systems

• There are different **types** of recommender systems
• They all **recommend** products/items...
• ...but use different **techniques** to find the best item(s)
• Three types of systems are **commonly** used
Content-based Systems

- Collect information about the items (e.g., keywords)
- Find items similar to ones the user liked in the past
- Idea: user preferences do not change
- Advantage: independent of other users
Content-based Systems

- Collect information about the items (e.g., keywords)
- Find items similar to ones the user liked in the past
Content-based Systems

- Collect information about the items (e.g., keywords)
- Find items similar to ones the user liked in the past
- Idea: user preferences do not change
Content-based Systems

- Collect information about the items (e.g., keywords)
- Find items similar to ones the user liked in the past
- Idea: user preferences do not change
- Advantage: independent of other users
Collaborative Filtering Systems

• Collect information about the user
Collaborative Filtering Systems

• Collect information about the user
• Find similar users
Collaborative Filtering Systems

• Collect information about the user
• Find similar users
• Idea: people who liked the same things will like the same in the future
Collaborative Filtering Systems

- Collect information about the user
- Find similar users
- Idea: people who liked the same things will like the same in the future
- Advantage: no understanding of the items necessary
• Explicit information about the *items* and *user*
Knowledge-based Systems

- Explicit information about the items and user
- Find items that fulfill the user-given constraints
Knowledge-based Systems

- Explicit information about the items and user
- Find items that fulfill the user-given constraints
- Idea: recommendation boils down to a constraint satisfaction problem
Knowledge-based Systems

- Explicit information about the items and user
- Find items that fulfill the user-given constraints
- Idea: recommendation boils down to a constraint satisfaction problem
- Advantage: no history of the user is necessary
A Generic Framework
Web-based Client-Server Model

- Subdivided into **frontend** and **backend**
• Subdivided into **frontend** and **backend**
• Backend is based on the Spring Framework (**Java**)
Web-based Client-Server Model

• Subdivided into **frontend** and **backend**
• Backend is based on the Spring Framework (**Java**)  
• Frontend is mobile-friendly **HTML5**
Web-based Client-Server Model

- Subdivided into frontend and backend
- Backend is based on the Spring Framework (Java)
- Frontend is mobile-friendly HTML5
- Parts are loosely coupled
Message Passing

Client

Server
Message Passing

Client

MessageHub
(JavaScript)

MessageHub
(Java)

Server
Message Passing

Client

Handler
Handler
Handler

MessageHub (JavaScript)

Server

Handler
Handler
Handler

MessageHub (Java)
Message Passing

Client

Handler

Handler

Handler

MessageHub (JavaScript)

Server

Handler

Handler

Handler

MessageHub (Java)

Message

type (string)
content (JSON)
Message to register a new user

```json
{
    type: "register",
    content: {
        username: "michael",
        password: "12345678",
        email: "michael.schwarz@noreply.com"
    }
}
```
Multiple Frontends

- Loose coupling and easy API allows easy implementation of new frontends
Multiple Frontends

- Loose coupling and easy API allows easy implementation of new frontends
- Bachelor Thesis: Implementation of a native iOS client
Knowledge Acquisition
Acquire Knowledge

• Users do not like lengthy tasks
Acquire Knowledge

• Users do not like lengthy tasks
• Acquire knowledge from the user using small tasks (microtasks)
Acquire Knowledge

- Users do not like lengthy tasks
- Acquire knowledge from the user using small tasks (microtasks)
- Microtask has only one question
Acquire Knowledge

- Users do not like lengthy tasks
- Acquire knowledge from the user using small tasks (microtasks)
- Microtask has only one question
- 6 different types of microtasks
Microtask #1

Item’s support regarding one specific attribute
Microtask #2

Best matching item regarding one specific attribute

Which item fits the answer «Museums» of the attribute «Sights» better?

- Paris
- Mumbai

Museums

0% 100%

Don't show questions for this recommender
Microtask #3

Best matching answer regarding one specific attribute

City

Item: Berlin. Which answer fits the attribute Activities best?

- Nightlife
- Shopping
- Dining
- Hiking
- Swimming

How well?

? 0% 100%

Don't show questions for this recommender
Microtask #4

Weighted answers regarding one specific attribute
Microtask #5

Implicit CAPTCHA

Which item belongs to the recommender »City«?

- [ ] Don’t show questions for this recommender
  - [x] Skip
  - [✓] Next
Microtask #6

Binary decision

Does the item "Beijing" belong to the recommender "City"?  

- Yes
- No

Don't show questions for this recommender

Skip  Next
Quality Assurance
• Users have to “earn” trust
Human Score

• Users have to “earn” trust
• Score is influenced by CAPTCHAs, user behavior, etc.
Human Score

- Users have to “earn” trust
- Score is influenced by CAPTCHAs, user behavior, etc.
- All contributions of the user are weighted with this score (0% - 100%)
• Users have to “earn” trust
• Score is influenced by CAPTCHAs, user behavior, etc.
• All contributions of the user are weighted with this score (0% - 100%)
• New or malicious users have minor to no influence on the knowledge base
• Depending on the human score, users get microtask with known answers (ground truth)
• Depending on the human score, users get microtask with known answers (ground truth)
• Similar to CAPTCHAs, but not seen as such by the user
• Depending on the human score, users get microtask with known answers (ground truth)
• Similar to CAPTCHAs, but not seen as such by the user
• Influence the human score (positively and negatively)
Depending on the human score, users get microtask with known answers (ground truth).
Similar to CAPTCHAs, but not seen as such by the user.
Influence the human score (positively and negatively).
Classify an image, hard to do automatically.
Timing Models

• We model the time it takes to answer a microtask
Timing Models

• We model the time it takes to answer a microtask
• Timings are matched using Kullback-Leibler distance
Timing Models

- We model the time it takes to answer a microtask
- Timings are matched using Kullback-Leibler distance
- Answers are weighted according to how well they fit
Timing Models

- We model the time it takes to answer a microtask
- Timings are matched using Kullback-Leibler distance
- Answers are weighted according to how well they fit
- Non-matching timings are discarded and decrease the human score
Microtask Timings

Type 1, $\mu = 1.5203, \ \sigma = 1.0814$

Type 2, $\mu = 1.8542, \ \sigma = 0.75089$

Type 3, $\mu = 1.7937, \ \sigma = 0.84687$

Type 4, $\mu = 1.8325, \ \sigma = 0.67965$

Type 5, $\mu = 1.6436, \ \sigma = 0.9707$

Type 6, $\mu = 1.3422, \ \sigma = 1.2493$
• Users can add new item, we have to cope with spam
• Users can add new item, we have to cope with spam
• CAPTCHAs only prevent automated spam
• Users can add new item, we have to cope with spam
• CAPTCHAs only prevent automated spam
• For a new item, we generate verification microtasks
• Users can add new item, we have to cope with spam
• CAPTCHAs only prevent automated spam
• For a new item, we generate verification microtasks
• If the community decides that an item does not belong to the recommender, it is removed
Data Collection

- We need knowledge for new items
Data Collection

- We need *knowledge* for new items
- Dynamic approach to calculate number of distributed microtasks
Data Collection

- We need knowledge for new items
- Dynamic approach to calculate number of distributed microtasks
- Loosely based on local working set algorithm for task scheduling
Data Collection

- We need **knowledge** for new items
- Dynamic approach to calculate number of distributed microtasks
- Loosely based on **local working set** algorithm for task scheduling
- Settle on minimum number of microtasks based on quality of the results
Evaluation
• We conducted a worldwide study
• We conducted a **worldwide study**
• Users had to complete microtasks, evaluate items, and use the recommender
• We conducted a worldwide study
• Users had to complete microtasks, evaluate items, and use the recommender
• 1307 users (90.9\%) completed all tasks
Study

- We conducted a **worldwide study**
- Users had to complete microtasks, evaluate items, and use the recommender
- 1307 users (90.9%) completed all tasks
- Quality assurance led to recommendation *improvement of >20%*
Recommendation Quality Improvement

![Graph showing QA improvement over raw data in % vs top n items considered]
Conclusion
• We developed a **generic recommender framework** for knowledge-based recommenders
Summary

• We developed a generic recommender framework for knowledge-based recommenders
• We showed that users are willing to contribute through small tasks
• We developed a **generic recommender framework** for knowledge-based recommenders
• We showed that users are willing to contribute through **small tasks**
• We presented automatic ways to ensure the **quality** of user content
Human Computation Based Acquisition of Financial Service Advisory Practices

Alexander Felfernig, Michael Jeran, Martin Stettinger, Thomas Absenger, Thomas Gruber, Sarah Haas, Emanuel Kirchengast, Michael Schwarz, Lukas Skofitsch, Thomas Ulz

FINREC’15
Peopleviews: Human computation for constraint-based recommendation

Alexander Felfernig, Thomas Ulz, Sarah Haas, Michael Schwarz, Stefan Reiterer, Martin Stettinger

ACM RecSys 2015 CrowdRec Workshop
Human computation for constraint-based recommenders

Thomas Ulz, Michael Schwarz, Alexander Felfernig, Sarah Haas, Amal Shehadeh, Stefan Reiterer, Martin Stettinger

Journal of Intelligent Information Systems 2016
A Short Overview of the PeopleViews Mobile User Interface

Angela Promitzer, Alexander Felfernig, Michael Schwarz, Thomas Ulz, Amal Shehadeh, Sarah Haas

Thank you for your attention!
In conclusion, AAAAAAAAAAAA!!!

The best thesis defense is a good thesis offense.
Recommendation Quality Improvement without Ground Truth
## Human Score Calculation Example

<table>
<thead>
<tr>
<th>User</th>
<th>Human Score</th>
<th>Answer 1</th>
<th>Answer 2</th>
<th>Answer 1 (weighted)</th>
<th>Answer 2 (weighted)</th>
</tr>
</thead>
<tbody>
<tr>
<td>User 1</td>
<td>1</td>
<td>0.8</td>
<td>0.3</td>
<td>0.8</td>
<td>0.3</td>
</tr>
<tr>
<td>User 2</td>
<td>0.5</td>
<td>0.9</td>
<td>0.4</td>
<td>0.45</td>
<td>0.2</td>
</tr>
<tr>
<td>User 3</td>
<td>0.5</td>
<td>0.6</td>
<td>0.5</td>
<td>0.3</td>
<td>0.25</td>
</tr>
<tr>
<td>User 4</td>
<td>0</td>
<td>0.2</td>
<td>0.7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>Sum</strong></td>
<td><strong>2</strong></td>
<td><strong>2.5</strong></td>
<td><strong>1.9</strong></td>
<td><strong>1.55</strong></td>
<td><strong>0.75</strong></td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>-</td>
<td>$\frac{2.5}{4} = 0.625$</td>
<td>$\frac{1.9}{4} = 0.475$</td>
<td>$\frac{1.55}{2} = 0.775$</td>
<td>$\frac{0.75}{2} = 0.375$</td>
</tr>
</tbody>
</table>

**Table 1:** Four different users and their support values for Answer 1 and Answer 2.
Optimal Number of Microtasks Example

<table>
<thead>
<tr>
<th>Cycle</th>
<th># of microtasks</th>
<th>Answered</th>
<th>Data is good</th>
<th>New # of microtasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycle 1</td>
<td>10</td>
<td>4</td>
<td>no</td>
<td>$10 \times 1.5 = 15$</td>
</tr>
<tr>
<td>Cycle 2</td>
<td>15</td>
<td>11</td>
<td>yes</td>
<td>$15 \times 0.75 = 11$</td>
</tr>
<tr>
<td>Cycle 3</td>
<td>11</td>
<td>6</td>
<td>yes</td>
<td>$11 \times 0.75 = 8$</td>
</tr>
<tr>
<td>Cycle 4</td>
<td>8</td>
<td>4</td>
<td>no</td>
<td>$8 \times 1.5 = 12$</td>
</tr>
</tbody>
</table>

Goal: 5 answers

**Cycle 1** Start with 10 tasks → not enough, increase to 15
## Optimal Number of Microtasks Example

<table>
<thead>
<tr>
<th>Cycle</th>
<th># of microtasks</th>
<th>Answered</th>
<th>Data is good</th>
<th>New # of microtasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycle 1</td>
<td>10</td>
<td>4</td>
<td>no</td>
<td>$10 \times 1.5 = 15$</td>
</tr>
<tr>
<td>Cycle 2</td>
<td>15</td>
<td>11</td>
<td>yes</td>
<td>$15 \times 0.75 = 11$</td>
</tr>
</tbody>
</table>

Goal: 5 answers

- **Cycle 1**: Start with 10 tasks → not enough, increase to 15
- **Cycle 2**: 15 was enough, decrease to $15 \cdot 0.75 = 11$ tasks
## Optimal Number of Microtasks Example

<table>
<thead>
<tr>
<th>Cycle</th>
<th># of microtasks</th>
<th>Answered</th>
<th>Data is good</th>
<th>New # of microtasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycle 1</td>
<td>10</td>
<td>4</td>
<td>no</td>
<td>$10 \times 1.5 = 15$</td>
</tr>
<tr>
<td>Cycle 2</td>
<td>15</td>
<td>11</td>
<td>yes</td>
<td>$15 \times 0.75 = 11$</td>
</tr>
<tr>
<td>Cycle 3</td>
<td>11</td>
<td>6</td>
<td>yes</td>
<td>$11 \times 0.75 = 8$</td>
</tr>
</tbody>
</table>

**Goal:** 5 answers

**Cycle 1** Start with 10 tasks → not enough, increase to 15

**Cycle 2** 15 was enough, decrease to $15 \cdot 0.75 = 11$ tasks

**Cycle 3** 11 was enough, decrease to $11 \cdot 0.75 = 6$ tasks
## Optimal Number of Microtasks Example

<table>
<thead>
<tr>
<th>Cycle</th>
<th># of microtasks</th>
<th>Answered</th>
<th>Data is good</th>
<th>New # of microtasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycle 1</td>
<td>10</td>
<td>4</td>
<td>no</td>
<td>$10 \times 1.5 = 15$</td>
</tr>
<tr>
<td>Cycle 2</td>
<td>15</td>
<td>11</td>
<td>yes</td>
<td>$15 \times 0.75 = 11$</td>
</tr>
<tr>
<td>Cycle 3</td>
<td>11</td>
<td>6</td>
<td>yes</td>
<td>$11 \times 0.75 = 8$</td>
</tr>
<tr>
<td>Cycle 4</td>
<td>8</td>
<td>4</td>
<td>no</td>
<td>$8 \times 1.5 = 12$</td>
</tr>
</tbody>
</table>

**Goal:** 5 answers

- **Cycle 1** Start with 10 tasks → not enough, increase to 15
- **Cycle 2** 15 was enough, decrease to $15 \times 0.75 = 11$ tasks
- **Cycle 3** 11 was enough, decrease to $11 \times 0.75 = 6$ tasks
- **Cycle 4** 8 was not enough, increase to $8 \times 1.5 = 12$ tasks
Position of Chosen Item

- Position 1: 34%
- Position 2: 15%
- Position 3: 13%
- Position 4: 9%
- Position 5: 8%
- Position 6: 6%
- Position 7: 6%
- Position 8: 6%
- Position 9-11: 3%