Computing Effective Recommendations for Clusters of Tourists

Francesco Ricci
Information and Database Systems Engineering
Free University of Bozen-Bolzano
fricci@unibz.it
Content

- Preferences and choices
- Recommender systems – the *classical view*
- How can we compute more effective recommendations?
- Guessing the user utility function *vs* optimizing predictive precision
- A novel approach: from choices (movements in the physical space) to experienced utility

  - *Groups of users with similar behaviours may better reveal the hidden utility of observed choices.*
What we like may not be what we choose

... and a Recommender System should suggest even better choices!
Preferences and Choices

- **Preferences** (utility) influences choices and are hard to model and acquire (ratings, likes, ...)

- **Choices** are easier to observe (clicks, purchases, replies, bookmarks, movements) - they are *noisier* indications of preferences – they may not even match our preferences

- *Observed choices should be used to predict preferences but in RSs are mostly used to predict other choices*

- A **useful** recommendation is for a **novel** item that the user will:
  1. **Accept** to choose (persuasion – expected utility)
  2. **Like** to consume (experienced utility)
Knowing your goals / preferences

- "what do I want?" – depends on how a choice will make us feel
  - **Future**: what you expect an experience will make you feel is called *expected utility*
  - **Present**: The way an item (movie, travel, car) makes you feel in the moment is called *experienced utility*
  - **Past**: Once you had an experience (e.g. a movie), future choice will be based on what you remember about that: *remembered utility*.

An RS can better serve users if it correctly estimates the expected and experienced utility of candidate items.
Recommender Systems and Utilities

- **Expected utility** can be derived from user **online** activity: clicks, dwell time, *like* for an item or an option, eye movements, ...
  - *Examples*: “like” for a destination image in Instagram, click on an offer in Booking

- **Experienced utility** can be derived from user **online** and **offline** activity: check in, post content, *like* for an item
  - *Examples*: post a picture on Instagram while visiting a place, comment an event on Twitter while attending it

- **Remebered utility**: can be derived from user **online** activity: rate or review an item
  - *Examples*: rating and reviewing a destination in Booking
Classical Recommendation Model

Three types of entities: **Users, Items and Contexts**

1. A **background knowledge**:  
   - A set of **ratings – preferences**  
     - \( r: \text{Users} \times \text{Items} \times \text{Contexts} \rightarrow \{1, 2, 3, 4, 5\} \)  
   - A set of “features” of the Users, Items and Contexts

2. A method for **predicting** the function \( r \) where it is unknown:  
   - \( r^*(u, i, c) = \text{Average ratings } r(u', i, c'): \text{users } u' \text{ are similar to } u \text{ and context } c' \text{ is similar to } c \)

3. A method for **selecting** the items to recommend (choice):  
   - In context \( c \) recommend to \( u \) the item \( i^* \text{ with the largest predicted rating } r^*(u,i,c) \)
Context Aware RS Algorithms

- Reduction-based Approach, 2005
- Exact and Generalized Pre Filtering, 2009
- Item Splitting, 2009
- Tensor Factorization, 2010
- User Splitting, 2011
- Context-aware Matrix Factorization, 2011
- Factorization Machines, 2011
- Differential Context Relaxation, 2012
- Differential Context Weighting, 2013
- UI splitting, 2014
- Similarity-Based Context Modelling, 2015
- Convolutional Matrix factorization, 2016
- Contextual bandit, 2018

The research has focussed mostly on context-dependent preference prediction rather than choice modelling and support.
Predicting Choices/Actions

- More recent models - based on user action observations - predict choices (e.g. sequences of movie views)
- Ironically, they claim to be able to predict user preferences
- There is a tendency to confuse preferences and actions/choices
- Some models can combine actions and preferences [Lavee et al., 2019]

We should understand how users make choices in context in order to understand how a RS can better support them in identifying items that they will happily *choose* and *not regret*.
Travel Choice Criteria

- When is cost effective
- When is liked by people that likes what we like
- When is good for the full traveling group
- When we did not yet think about that
- When is not what we did last year
- When it has the features that we usually like
- When it has some impressive features
- When it is much better than other options
- When it is similar to what we did previous years
- When the weather will be great.
Do you believe that by simply mining a data set of users’ ratings (or choices) one can generate useful travel recommendations?

We need to better understand and structure the knowledge that can be derived from the data.

We need to better understand the current user’s relevant context and utility function.
Marschall McLuhan

“The medium is the message”
Recommendation Lists

List 1
- Uffizi Galleries
- Michelangelo Square
- Duomo

List 2
- San Marco Museum
- S. Croce Square
- Santa Maria Novella

TripAdvisor images
Recommendation Lists

- Uffizi Galleries
- Michelangelo Square
- Duomo

Expected utility can be manipulated

My selection of images

- San Marco Museum
- S. Croce Square
- Santa Maria Novella
Points of Interest

- A number of features and contextual factors influence both the \textit{pre-visit} evaluation and the \textit{experience} of a POI
  - Traveller’s knowledge of the place
  - What she has already visited and when
  - Pictorial representation
  - Distinguished features
  - Travel party
  - Previous knowledge/usage of the app/recsys (the medium)
  - Popularity, fashionableness, trendiness, fame, prominence, prestige, reputation, visibility, rank.
Expected vs. Experienced Utility

- Should the system optimize expected or experienced/remembered utility?
- Hence, should it use choice data or ratings/judgements data?

**Expected Utility**

+ It matches the **user goals** at decision time
  - It is subject to **bias** of user’s judgement
+ It is derived from an **unbiased sample** of observations
  - It is derived from observations without **meaning**.

**Experienced Utility**

+ It is an **explicit** user assessment
  - It is **incomplete** (data)
+ It depends on the consequence of choice (**context** is used)
  - It is often derived from **remembered** utility.
Understanding and Influencing Users

- We do not know which criteria/heuristics are influencing the expected/experienced utility of the user and their choices.

- The users may make wrong estimations: expected utility may be different from what they will actually experience.
  - Novel items are hard to assess for their value.

- We must make reasonable assumptions on the user’s important criteria in order to suggest relevant items by correctly leveraging the observation of user behaviour (model learning).

- ... and convince the user that this is a good choice (raise the expected utility).
Clustering People

- We have recently tried to learn/predict the (experienced) utility function of users with a technique that make use of clusters of users and inverse reinforcement learning.

- **Group** and **model travellers** with observable similar behaviour and optimize the recommendations for them – not purely individual recommendations.

Behaviour and Recommendation

- **Behaviour learning** and **recommendation** should be decoupled.

- An **exact behavioural model**, e.g., what points of interest a user is likely to visit may produce **not novel and uninteresting recommendations**.

- Recommendation should **optimize the criteria** that ultimately have determined the observed behaviour.
Behavioral Model Learning

- Learning user behaviour, but suggest to deviate from the usual behaviour

  - The user is predicted to take a coffee at 8:00 at Walter Bar
    - The system suggests to get coffee at Rosy Bar – it is cheaper and better

We must understand that the user likes good Italian and cheap coffee – not that he likes to go to Walter Bar at 8:00!
Grouping Travellers
Clustering Users’ Visit Trajectories

- One visit to Florence, *taking pictures of:*
  - Pitti Palace; Boboli Garden; Uffizi Museum

- Model a **POI** with important **keywords**, contextual conditions and expert-defined features

- Model a **trajectory** with **behavioural features**

- **Cluster trajectories:** a cluster models a group of users with similar behaviour.
Data and Clusters

<table>
<thead>
<tr>
<th>Dataset</th>
<th># POIs</th>
<th># Trajectories</th>
<th># Features</th>
<th>Context</th>
<th>Content</th>
<th>“Expert”</th>
<th>Behaviour</th>
</tr>
</thead>
<tbody>
<tr>
<td>Florence</td>
<td>316</td>
<td>2110</td>
<td>15</td>
<td>29</td>
<td>9</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Rome</td>
<td>376</td>
<td>4340</td>
<td>14</td>
<td>28</td>
<td>9</td>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

- **POI level – state model**
  - **Context:** weather (e.g., cloudy), temperature (e.g., hot), daytime (e.g., evening)
  - **Content:** POI’s categories extracted from TripAdvisor
  - **Expert:** excellence, POI’s reputation, ... (from TripAdvisor)

- **Trajectory level**
  - **Behaviour:** duration, #POIs visited, Avg. Dwell time, #POIs in top-n-TripAdvisor, #POIs TripAdvisor excellent
Clusters in Florence and Rome

Clustering uses only the behavioural features of the trajectory
Inverse Reinforcement Learning

- **Assumption:** the reward obtained by visiting a POI is determined by the POI’s **features** and the visit **context** – linear function of the POI’s vector

- Inverse Reinforcement Learning estimates the hidden reward function (experienced utility) that the users in a cluster apparently tried to maximise with the observed behaviour.

- We use Maximum Likelihood IRL.

- The users are assumed to better served if they choose, in state $s$, the visit action $a$ with the **largest immediate and future reward**: $Q(s, a)$ function.

Coping with Data Scarsity

- **State** \( s \) := the current position of the user (POI + Context)
- **Action** \( a \) := the next position of the user (POI + Context)
- *There are states \( s \) that have never been observed in the data (POIs in specific contextual conditions)*

**Q-BASEX recommendation**

- When \( Q(s, a) \) is not defined because the POI represented by \( s \) has never been observed in that **Context**: average \( Q \) over all the states with the same values of the **Content** features
- When \( Q(s, a) \) is not defined because the POI represented by \( s \) has **never been observed**: compute \( Q(s', a) \) for states \( s' \) representing **similar POIs** and take a weighted average.
Evaluation

- **Precision**: percentage of next POI correctly predicted
- **Reward**: difference in reward between the predicted next POI and the ground truth
- **Similarity**: how similar is the predicted next POI and the ground truth
- **I-Coverage**: average ratio of the POIs in the ground truth that are recommended
- **Unique**: ratio of the # of unique POIs recommended over the # of recommendations
- **Popularity**: ratio of the # of “popular” POI recommended over the # of recommendations.

**Train** IRL model on 80% and **Test** on 20% of the available trajectories in each cluster

Test trajectories are split in: 70% observed and 30% ground truth.
Results

- **SKNN** strong session-based knn algorithm – scores a next POI with the sum of the cosine similarities of the target trajectory and the neighbours trajectories containing the next POI.

<table>
<thead>
<tr>
<th>Model</th>
<th>Top-n</th>
<th>Prec</th>
<th>Rew</th>
<th>Sim</th>
<th>I-Cov</th>
<th>Unique</th>
<th>Pop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q-BASEX</td>
<td>1</td>
<td>0.10</td>
<td>0.44</td>
<td>0.10</td>
<td>0.33</td>
<td>0.36</td>
<td>0.79</td>
</tr>
<tr>
<td>SKNN</td>
<td>1</td>
<td>0.02</td>
<td>-0.03</td>
<td>0.06</td>
<td>0.28</td>
<td>0.30</td>
<td>0.88</td>
</tr>
<tr>
<td>Q-BASEX</td>
<td>3</td>
<td>0.09</td>
<td>0.26</td>
<td>0.10</td>
<td>0.53</td>
<td>0.21</td>
<td>0.78</td>
</tr>
<tr>
<td>SKNN</td>
<td>3</td>
<td>0.06</td>
<td>0.00</td>
<td>0.08</td>
<td>0.38</td>
<td>0.14</td>
<td>0.92</td>
</tr>
</tbody>
</table>

Florence data set
Why optimizing precision is bad

- If we optimize for precision the system will learn to recommend the items that the user found autonomously – not «useful» recommendations
- When the precise recommendations are \textit{finished} (already recommended) the system is unable to find novel recommendations
- Measured precision is typically very low (10\% in our data set) – so the user is mostly exposed to imprecise recommendations.

Advantages of the IRL method

- It is based on sequence mining but it can also generalise and suggest items never consumed before – depending on the chosen item features and context.

- By grouping users it can fix the “errors” of single users choosing apparently good items (wrong maximization of the experienced utility).

- Recommendations are not the predicted actions, they are the “optimal” ones (which could be further corrected if we better know the user utility function).
Lesson Learned

- Distinction between preferences and choices
- Use “knowledge” for modeling preferences and choices
- The distinction between expected and experienced utility and their role in recommendations
- Individual recommendation may be generated by assuming that groups of similar user are driven by a hidden utility function
  - But we must make good assumptions on
    - How the groups are clustered
    - The criteria in the utility function
Thanks

... in particular to my students and collaborators who contributed to develop these ideas:

- David Massimo
- Linas Baltrunas
- Laura Bledaite
- Marius Kaminskas
- Marko Gasparic
- Marko Tkalcic
- Matthias Braunhofer
- Mehdi Elahi
- Saikishore Kalloori
- Tural Gurbanov
- Thuy Ngoc Nguyen