Context and Intention-Awareness in POIs Recommender Systems

Hernani Costa
CISUC, University of Coimbra
Coimbra, Portugal
hpcosta@dei.uc.pt

Barbara Furtado
CISUC, University of Coimbra
Coimbra, Portugal
bfurtado@student.dei.uc.pt

Durval Pires
CISUC, University of Coimbra
Coimbra, Portugal
durval@student.dei.uc.pt

Luis Macedo
CISUC, University of Coimbra
Coimbra, Portugal
macedo@dei.uc.pt

Amilcar Cardoso
CISUC, University of Coimbra
Coimbra, Portugal
amilcar@dei.uc.pt

ABSTRACT

This paper describes an agent-based approach for making context and intention-aware recommendations of Points of Interest (POI). A two-parted agent architecture was used, with an agent responsible for gathering POIs from a location-based mobile application, and a set of Personal Assistant Agents (PAA), collecting information about the context and intentions of its respective user. Each PAA includes a probabilistic classifier for making recommendations given its information about the user's context and intentions. Supervised, incremental learning occurs when the feedback of the true relevance of each recommendation is given by the user to his PAA. To evaluate the system's recommendations, we performed an experiment based on the profile used in the training process, using different locations, contexts and goals.

Keywords

Context, Information Overload, Machine Learning, Personal Assistant Agents, Points of Interest Recommendation.

1. INTRODUCTION

With the technological advance registered in the last decades, there has been an exponential growth of the information available. In order to cope with this superabundance, Recommender Systems (RS) are a promising technique to be used in location-based systems (see [13, 4]). Most existing RS' approaches focus on either finding a match between an item's description and the user's profile (Content-based [2, 12, 10]), or finding users with similar tastes (Collaborative Filtering [8, 5, 6]). These traditional RS consider only two types of entities, users and items, and do not put them into a context when providing recommendations. Nevertheless, the most relevant information for the user may not only depend on his preferences, but also in his context. In addition, the very same content can be relevant to a user in a particular context, and completely irrelevant in a different one. For this reason, we believe that it is important to have the user's context in consideration during the recommendation process [14, 1]. Such systems can be useful in POIs RS [4, 7, 3].

In this paper, we intend to analyse the advantages of using a Multiagent System (MAS) capable of filtering irrelevant information, while taking into account the user's context. Our system uses standard POI attributes, and also integrates dynamic context data like user's context and goal, in order to process the requests. The system is able to understand the differences between each user, since each one of them has unique preferences, intentions and behaviours, resulting in different recommendations for different users, even if their context is the same.

The remaining of the paper starts with a presentation of the system's architecture (section 2). In section 3, we present the experimentation performed. Finally, section 4 presents our conclusions.

2. SYSTEM ARCHITECTURE

In this section, we present the system's architecture and all its components (see figure 1).

This architecture can be seen as a *middleware* between the user's needs and the information available in our system. More specifically, the Master Agent is responsible for starting, not only the agents (described in figure 1 as Agent_1 $\cdots$ Agent_n) that gather data from the Web resources (i.e., location-based mobile applications), but also the user's Personal Assistant Agent (PAA). Moreover, it is capable of aggregating the POIs returned from the Web agents into a well-defined knowledge representation.

The main purpose of each Web agent is to obtain all the POIs' information available in pre-defined Web sources. These autonomous agents are constantly searching for new information, and verifying if the data stored in the database (presented in figure 1 as POIs Database) is up-to-date.

As we can see in figure 1, each user has a PAA assigned to him. This agent expects a request from the user, and, based on his context, recommends a list of nearby POIs (see section 3). The PAA learns from the user's past experiences, in order to improve its recommendations. Specifically, a

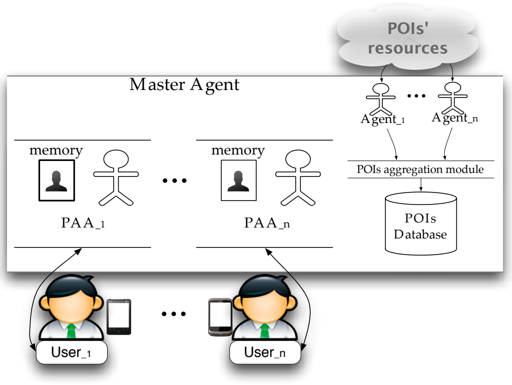

Figure 1: System's Architecture.

probabilistic classifier is used for that purpose, i.e., the PAA assigns a probability value to the relevance of the POI, given its information, the current user's context and intentions. Therefore, when the feedback of the true relevance of each recommendation is given by the user to his PAA, the PAA updates its memory. As a result, the agent learns every time the user decides to make a request and give his feedback.

3. EXPERIMENTAL WORK

Our main goal is to show that we can face the problem of location-based context-aware recommendations with a MAS architecture. In addition, we intend to verify how machine learning algorithms suit the task of predicting the user's preferences, based on his context. An effectiveness evaluation of our RS, in terms of the accuracy of its predictions, will be performed. This section presents the experimentation, in a controlled simulation, carried out to study the system's performance while recommending POIs. Firstly, the experiment set-up is presented (see 3.1), followed by an exhaustive analysis of the results (see 3.2).

3.1 Experiment Set-up

Our MAS contains agents responsible for obtaining POIs from Web sources. The purpose of these agents is to keep the information up-to-date in the database (see 3.1.1). On the other side, the system has PAAs, that use memory to save the user's experiences (see 3.1.2). In this experiment, only one information source was used (see 3.1.1) and only one user's profile (see 3.1.3). The experimentation was performed in a specific area of the city of Coimbra (Portugal), explained in detail in 3.1.5. Different scenarios were used to specify both the user's and POIs' contexts (see 3.1.4). To evaluate the system, some well-known metrics are presented in 3.1.6.

3.1.1 Agent Gowalla

As previously mentioned, our system could receive input from various location-based applications. In this experiment in particular, it is used one of the existing POIs' sources available on the Web, which explains why we used only one agent to obtain POI information.

Agent Gowalla obtains all the information through calls to Gowalla's API. It starts by requesting for POIs in a pre-defined area (see 3.1.4). During this process, it filters all the POIs that do not belong to the categories we will use (see 3.1.4). This process is repeated every 30 seconds, to allow the agent to detect new POIs whenever they are created, or to discover changes in an existing POI.

Due to the fact that Gowalla's database does not have all the information needed for the experiment, we decided to gather more information about the POIs on the field. This allowed us to have more details about each POI in order to fulfil the requirements of the experiment (see 3.1.5 for more details). After filtering the unused categories (irrelevant to this experiment), this extra information was combined with Gowalla's info in the aggregation module, being then saved to the database (see section 3.1.5).

3.1.2 Dataset

The recommendation process resorts to WEKA's API[1]. In order to predict if a POI would be useful for the user and if its recommendation is worthy, it was used a probabilistic classifier that was trained with the Naive Bayes Updateable algorithm. The predicted values vary from 1 (totally irrelevant) to 5 (most relevant), and the algorithm automatically distributes the probability ranges in this scale. POIs with a classification of at least 3, are recommended to the user.

When an agent recommends POIs to its user, the agent expects the user to rate each recommendation, and saves this information into its memory, which allows it to learn from the experience. The agent's memory is a set of instances, which we call dataset. In table 1, we can see an example of a dataset. The first five columns correspond to the information related to the POI: ID, category, price, schedule (morning, afternoon and night) and day off. The distance field corresponds to the distance between the POI and the user (near, average or far). The following three columns correspond to the user's context information: time of day, day of the week and his current goal (coffee, lunch, dinner or party). The last column (Label), corresponds to the algorithm's prediction.

3.1.3 User's Profile

As explained in section 2, each user has his own PAA (i.e., a dataset with his own preferences). We performed a simulation period in order to train the PAAs' classifiers. Since we had various PAAs' classifiers (each one with different user's profiles), it was impossible to evaluate all of them and we had to choose only one profile. This profile can be seen as a stereotype of a user who prefers POIs that are near, cheaper and not closed. For the sake of clarity, the feedback given by the user only considers the POIs' categories and not their names.

3.1.4 Definition of Scenario

Scenario is defined as the set of information related to the user which a PAA needs to classify a POI, in a certain context. More precisely, a scenario results from the combination of the user's context with the POI's context. We have defined the user's context by: i) proximity related to a specific POI `(far, average or near, where we consider near`$\leq$`100m, 100m<average`$\leq$`200m and far>200m)`[2] ; ii) current time of day `(morning, afternoon and night)`; iii) current day of the week; iv) user's goal

[1]http://weka.sourceforge.net/doc

[2]It were used "small distance amplitudes" because in this

Table 1: Dataset example.

POI id	Category	Price	Schedule	DayOff	Distance	TimeOfDay	DayOfTheWeek	Goal	Label
7086048	Bakery	Cheap	Morning/Afternoon/Night	Sunday	Average	Night	Saturday	Coffee	5
7023528	Apparel	Cheap	Morning/Afternoon	Sunday	Far	Afternoon	Friday	Lunch	1
1512823	Pub	Cheap	Night	Sunday	Far	Night	Friday	Party	4

`(coffee, lunch, dinner and party)`. The POI's context is defined by the POI: a) `id`; b) `category`; c) price `(cheap, average, expensive)`; d) timetable `(morning, afternoon, night, or combinations)`; e) day off `(a day of the week or combinations)`.

3.1.5 Area of work

The number of POIs that exist in Coimbra (Gowalla returned about 954) made it impossible to manually evaluate the whole city. For this reason, it was used a smaller part of the city that had more POIs density and diversity (Coimbra's Downtown). So, we studied the type of POIs in that area, and also restricted the set to three main categories ({Food, Shopping, Nightlife}, the categories that contain more POIs). The number of sub-categories for Food are 44, Shopping 51 and Nightlife 11, with 59, 29 and 29 different POIs, respectively.

As referred above (see 3.1.1), we gathered more information about the POIs. The extra information we manually gathered from the places, was the POI's: Price `(cheap, average or expensive)`; DayOff `(day(s) the POI closes)`; Timetable `(part of the day in which the POI is open)`. So, the combination of this new data with Gowalla's information, fulfils the POI's context.

3.1.6 Metrics

In this topic we present the metrics that will be used in our experiment. Equation 1 will be used to correlate two different types of data. Precision, recall and F_1 formulas will be used to analyse the system's accuracy.

The **Correlation Coefficient** (ρ) is used to return the correlation coefficient between two arrays, m_i and x_i, where $\{m_i, x_i\} \in \mathbb{R}$, $\rho \in \mathbb{R}$: $-1 \leq \rho \leq 1$, being $i \in \mathbb{N}$ and corresponding to the matrix's index.

$$\rho(m_i, x_i) = \frac{\sum_i (m_i - \overline{m})(x_i - \overline{x})}{\sqrt{\sum_i (m_i - \overline{m})(x_i - \overline{x})}} \tag{1}$$

Precision will be used to evaluate the quality of the recommendations. Specifically, it is the number of correctly recommended POIs divided by the total number of recommended POIs.

$$Precision = \frac{Correctly\_recommended\_POIs}{Total\_recommended\_POIs} \tag{2}$$

Recall evaluates the quantity of POIs extracted. More precisely, it is the number of correctly recommended POIs, divided by the total number of correctly evaluated POIs that should have been retrieved.

$$Recall = \frac{Correctly\_recommended\_POIs}{Total\_correct\_POIs} \tag{3}$$

The $\boldsymbol{F}_1$ score can be interpreted as a weighted average of the precision and recall.

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4}$$

3.2 Results

Our experiment can be divided in two different evaluations: cross validation (3.2.1), and the use of metrics (3.1.6) to compare the output recommendations given by the system with manual evaluation (3.2.2 and 3.2.3). It is important to explain that the system's classifier was trained using a dataset (see 3.1.2) containing: the original training dataset (which has correct classifications given by us); and a list of instances that were created from all POIs the system recommended during the simulation period. These POIs that were recommended by the system were inserted in that dataset, not with the classification given by the system, but instead with the feedback given by the user during the simulation. The resulting classifier was used to do the cross validation experiment (3.2.1).

3.2.1 Cross Validation

It was chosen to do 10 runs and 10 folds, because this is a combination that guarantees better evaluation [11]. In table 2 we can verify the percentage of correctly and incorrectly classified instances, and check some statistics from our classifier's performance. The results show that in a total of 14616 instances, the classifier correctly classified 9246 (63%), which can been seen as a good start.

Table 2: Classifier's statistics.

Correctly Classified Instances	9246	63.2594%
Incorrectly Classified Instances	5370	36.7406%
Kappa statistic	0.3909	
Mean absolute error	0.1729	
Root mean squared error	0.3163	
Relative absolute error	73.0797%	
Root relative squared error	91.9724%	
Total Number of Instances	14616	

Table 3 shows the detailed accuracy of our classifier, by class (**Cl**). Each class corresponds to the prediction values, in a scale of 1 to 5, as explained in section 3.1.2. For each class, the table shows the percentage of true positive (**TP**), false positive (**FP**), precision (**P**), recall (**R**), F_1 score (**F**$_1$) and **ROC Area**. The results demonstrate that class 1 has better results. This is due to the greater number of instances in the training dataset, classified with 1. Indeed, this happens because in many user's contexts there are always some irrelevant POIs (for instance, POIs that do not suit the user's goal). This makes the classifier more accurate in this class. Although the remaining classes do not have the same accuracy, their results are also very promising.

3.2.2 Manual Evaluation

To test our approach, we used a set of pre-defined scenarios that simulate real situations. Although we only used three different user locations (the ones that had more

experiment we only considered situations in which the user reach his destination on foot.

Table 3: Cross validation's statistics.

TP	FP	P	R	F_1	ROC Area	Cl
0.717	0.283	0.745	0.717	0.731	0.745	1
0.584	0.416	0.443	0.584	0.504	0.885	2
0.552	0.448	0.410	0.552	0.470	0.914	3
0.413	0.587	0.490	0.413	0.448	0.816	4
0.489	0.511	0.630	0.489	0.550	0.957	5

POI density), we analysed 18 different user contexts (see section 3.1.4). This 18 combinations were named runs.

The 18 runs resulted from the combination between different user's request, each one in a specific context (see section 3.1.4) and all the nearby POIs recommended by the system. More precisely in this experiment, it was used three user's locations in six different situations. Goal, time of day, day of the week: `[Coffee, Morning, Sunday]; [Coffee, Afternoon, Monday]; [Coffee, Night, Tuesday]; [Lunch, Afternoon, Wednesday]; [Dinner, Night, Friday]; [Party, Night, Saturday]`.

Our goal was to compare the system's recommendations with a manual evaluation made by human judges, and to apply some metrics to analyse our system's performance.

The judges evaluated every POI, from every run, according to the current user's context and POI's context, using the following scale: `0` - if the POI does not satisfy the user's context or the user's goal; `1` - if the POI satisfies the user's context and the user's goal, but if it is expensive or too far from the user; `2` - if the POI satisfies the user's context and the user's goal, and it is not expensive or far.

It is important to refer that the classifier's training dataset was built based on the preferences of a particular user's profile (i.e., POIs that were near, cheaper, and that were not closed, see section 3.1.3 for more details). The evaluation performed by the human judges was also based on the preferences of the same user's profile. They were asked to give their personal opinion for a list of scenarios, but never contradicting the user's profile they were simulating.

To perform the manual evaluation, we create a user interface using Google Maps[3]. The POIs' names were omitted to avoid that the judges' personal opinion influenced the evaluation, since the classifier was trained based on the POI's category. We had to do this to prevent discrepancy between the judges preferences and the user's profile (3.1.3).

The manual evaluation was important to evaluate the performance of the system in ambiguous cases (a POI with an average price and average distance). In this specific situations, the agreement between the human judges was low (14.3%). However, the PAA was trained to a specific user's profile and it is expected, in these ambiguous cases, to give better results for the judge with preferences closer to the user's profile used in the training process (notice that each user has a PAA that learn with his preferences, individually).

Furthermore, the exact agreement among judges resulted in 93.3% (using the three values in the scale: {0, 1, 2}). In addition, we also calculated the relaxed agreement (using a scale of {0, 2}, considering POIs classified as 1 and 2 as correct), resulting in 95.7%.

3.2.3 Manual Evaluation vs. Automatic Recommendations

In order to observe the relationships between the manual evaluation and the output values given by the RS, the correlation coefficients between them were computed and are shown in figure 2. In addition, the F_1 score was calculated (figure 3). The x-axis represents a run, which corresponds to the simulation of a user's request in a specific context (see 3.1.4) and all the nearby POIs recommended by the system (see 3.2.2). We simulated different requests, leading to a total of 18 runs. More specifically, runs: {1, 2, 3, 7, 8, 9, 13, 14, 15} = goal Coffee; {4, 10, 16} = goal Lunch; {5, 11, 17} = goal Dinner; and {6, 12, 18} to the goal Party.

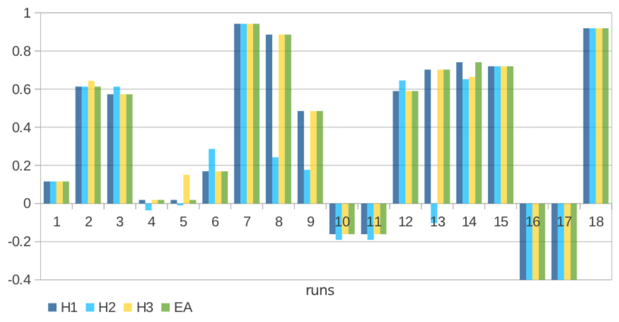

Figure 2: Correlation coefficients between manual evaluation (with exact agreement) and the system's recommendations.

In order to avoid some of the ambiguity that could arise when using a 1-5 scale, the human judges evaluated the system in a scale of 0 to 2 (see section 3.1.2 and 3.2.2, respectively). Furthermore, to calculate the correlation (eq. 1) between the system's recommendations and the evaluation of the human judges, the scale of both evaluations was standardised. The system's scale was converted to a scale from 0 to 2: where 1 and 2 corresponds to 0; 3 corresponds to 1; and 4 and 5 corresponds to 2. Therefore, figure 2 shows the correlation coefficients between the most common evaluated value (i.e., the exact agreement correlation, represented as **EA**) given by each of the human judges (corresponding to **H1**, **H2** and **H3**, in the chart) and the system's recommendations, through the 18 runs.

As we can see in figure 2, the results are promising. However, some of the results have low correlation values because when we trained the system, we discarded all contexts that make no sense, like having lunch at night or morning and having dinner or to party at morning or afternoon. On the other hand, the goal Coffee is valid in all times of day, resulting in a lot more instances and, consequently, the system performed better when this was the user's goal. In order to overcome this problem, more instances with goals Lunch, Dinner and Party should be added to the training dataset.

The figure 3 shows the evolution of the F_1 (eq. 3) values (y-axis), in all the 18 runs (x-axis). In the figure 3, the results represented by the legends named **High** and **Low**, correspond to recommendations given by the system, with a score of 2 and a score of 2 and 1, respectively. This allow us to compare the results with a high filter, considering only the best recommendations (score 2), and with a low filter, considering all the good recommendations (score 1 and 2).

As we expected, higher values are obtained for the goal Coffee (see for example run 7 and 8), and low values are obtained for the goal Lunch and Dinner (see for instance runs 16 and 17). This happens because, as we mentioned

[3]http://code.google.com/apis/maps/index.html

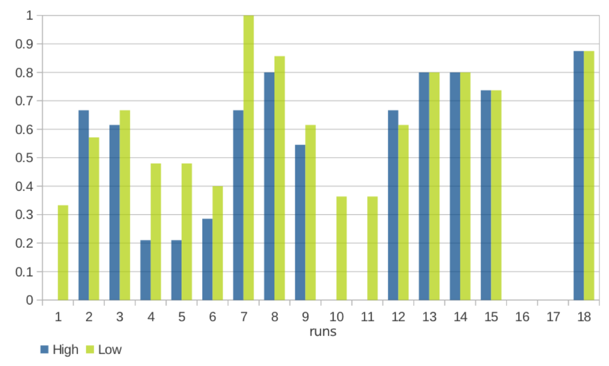

Figure 3: F-Measure.

before, the goal Coffee is valid in all times of day and the system performed better in these situations.

4. CONCLUSIONS

In this paper, we discussed the combination of context and intention-awareness with RS, applied in a location-based application. We pointed out what advantages are earned in using, besides the context, the user's intentions, and how to integrate both into a location-based RS. We also presented our system's architecture and described its advantages, such as its modular nature. Machine learning techniques were used to train the classifiers, more precisely the Naive Bayes Updateable algorithm. Machine learning can be a powerful tool to predict which content will be interesting for a determined user, but it should be used with caution and the datasets must be well defined.

Then, we created an experimental set-up to evaluate the system's performance. To test the accuracy of our system, we used various evaluation methods. First, we did a cross-validation test. Next, in order to observe the relationships between the manual evaluation and the output values given by the PAA, the correlation coefficients between them were computed. Nevertheless, after analysing the results in general, the recommendations can be considered very promising, being this a good starting point to develop a context and intention-aware POI RS.

In the future, we are planning numerous improvements to our work, such as: take into account new attributes (e.g., POI's quality); test and compare other machine learning algorithms; analyse other users' profiles; use new information sources; and make it available to the community in order to get more feedback. We think that with more data and more training the results from our system could improve. Furthermore, we intend to make available to the user the possibility of changing what values fit in each attributes (e.g., what price is considered cheap). Moreover, we plan to analyse the system accuracy when applying selective attention metrics, such as surprise [9], in the recommendation outputs.

Acknowledgments

Work funded by Fundação para a Ciência e Tecnologia — Project PTDC/EIA-EIA/108675/2008, and by FEDER through Programa Operacional Factores de Competitividade do QREN — COMPETE:FCOMP-01-0124-FEDER-010146.

5. REFERENCES

[1] G. Adomavicius, B. Mobasher, F. Ricci, and A. Tuzhilin. Context-Aware Recommender Systems. *AI Magazine*, 32(3):67–80, 2011.

[2] M. Balabanović and Y. Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, 1997.

[3] L. Baltrunas, B. Ludwig, S. Peer, and F. Ricci. Context relevance assessment and exploitation in mobile recommender systems. *Personal and Ubiquitous Computing*, 16(5):507–526, 2012.

[4] C. Biancalana, A. Flamini, F. Gasparetti, A. Micarelli, S. Millevolte, and G. Sansonetti. Enhancing Traditional Local Search Recommendations with Context-Awareness. In *User Modeling, Adaption and Personalization*, pages 335–340. Springer, 2011.

[5] D. Billsus and M. J. Pazzani. Learning Collaborative Information Filters. In *Proc. 15th Int. Conf. on Machine Learning*, pages 46–54, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[6] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proc. 16th Int. Conf. on World Wide Web*, pages 271–280, New York, NY, USA, 2007. ACM.

[7] H. Huang and G. Gartner. Using Context-Aware Collaborative Filtering for POI Recommendations in Mobile Guides. In *Advances in Location-Based Services*, Lecture in Geoinformation and Cartography, pages 131–147, Vienna, Austria, 2012. Springer.

[8] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl. GroupLens: applying collaborative filtering to Usenet news. *Commun. ACM*, 40(3):77–87, 1997.

[9] L. Macedo. A Surprise-based Selective Attention Agent for Travel Information. In *Proc. 9th Int. Conf. on Autonomous Agents and Multiagent Systems, 6th Workshop on Agents in Traffic and Transportation*, pages 111–120, 2010.

[10] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proc. 18th National Conf. on AI*, pages 187–192, Menlo Park, CA, USA, 2002. AAAI.

[11] P. Refaeilzadeh, L. Tang, and H. Liu. Cross-Validation. In *Encyclopedia of Database Systems*, pages 532–538. Springer, 2009.

[12] W. van Meteren and M. van Someren. Using Content-Based Filtering for Recommendation. In *Proc. ECML/MLNET Workshop on Machine Learning and the New Information Age*, pages 47–56, Barcelona, Spain, 2000.

[13] M. van Setten, S. Pokraev, and J. Koolwaaij. Context-Aware Recommendations in the Mobile Tourist Application COMPASS. In *Proc. 3rd Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems*, pages 235–244, Berlin, 2004. Springer.

[14] W. Woerndl and J. Schlichter. Introducing context into recommender systems. In *Proc. AAAI, Workshop on RS in e-Commerce*, pages 22–23, Vancouver, Canada, 2007.