Harnessing Crowdsourced Recommendation Preference Data from Casual Gameplay

Barry Smyth, Rachael Rafter, Sam Banks
Insight Centre for Data Analytics
University College Dublin, Dublin, Ireland
firstname.surname@insight-centre.org

ABSTRACT

Recommender systems have become a familiar part of our online experiences, suggesting movies to watch, music to listen to, and books to read, among other things. To make relevant suggestions, recommender systems need an accurate picture of our preferences and interests and sometimes even our friends and influencers. This information can be difficult to come by and expensive to source. In this paper we describe a game-with-a-purpose designed to infer useful recommendation data as a side-effect of gameplay. The game is a simple, single-player matching game in which players attempt to match movies with their friends. It has been developed as a Facebook app and harnesses the social graph and likes of players as a source of game data. We describe the basic game mechanics and evaluate the utility of the recommendation knowledge that can be inferred from its gameplay as part of a live-user trial.

1. INTRODUCTION

Recommender systems have become a familiar part of our online experiences, suggesting products, items and services from our favourite online stores to entertainment and news sites. Today recommender systems have an influence on what we read, listen to, and watch. They often determine where we vacation and may even influence our choice of a mate.

To make good suggestions recommender systems use various types of information. Most rely on user preferences, such as item ratings [1]. Many take advantage of matrix factorisation methods to find hidden patterns within these preferences [19]. Others harness inter-user similarity to identify groups of like-minded users [7]. Some even leverage social network information to infer trust or influence relationships between users [11, 12, 27, 28]. The source of this information, and the ability to collect it at scale for many millions of users, has been the subject of much research within the recommender systems community. Many approaches have been considered, from using explicit feedback such as transaction histories to inferring interest from indirect signals such as read-times or sharing [17]. It is always interesting to consider novel ways to collect these types of data.

GWAPs (games-with-a-purpose) are casual computer games that are typically simple and fun to play. They are designed so that gameplay contributes to some secondary problem solving goal; e.g. the ESP Game invites pairs of players to guess words for a specific image [40]. Players gain points when they guess the same words and as many players compete on the same images their guesses contribute to a rich tag-based representation of the images. Similar games have been used to describe other forms of media such as audio and video [10,22,37] and more sophisticated GWAPs have been developed with other tasks in mind, from object segmentation [31] to protein folding [6].

The power of a GWAP stems from its ability to attract large numbers of players each of whom contributes some fragment of solution knowledge as part of a greater goal through their natural gameplay. GWAPs take advantage of tried and trusted game mechanics to offer players a gaming experience that is compelling and fun, often attracting large numbers of players to harness considerable collective intelligence. If GWAPs can be used for challenging tasks such as object recognition and protein folding might they also be used to help build better recommender systems?

This is the question that motivates our work. In particular, we will describe a GWAP designed to infer useful recommendation knowledge as a side-effect of gameplay. The paper builds on initial work presented in [3] which proposes a simple, single-player matching game in which players attempt to match movies with their friends. In this paper we describe how these matches can be used to infer the strength of relationships between users (an important source of knowledge for many recommender systems) as well as the likely level of interest a user will have in a particular movie (another key source of recommendation knowledge). The game has been developed as a Facebook app and harnesses the social graph and likes of players as a source of game data. In this paper we describe the basic game mechanics in detail and evaluate the utility of the recommendation knowledge that can be inferred from its gameplay as part of a live-user trial.

2. RELATED WORK

This work brings together ideas from the fields of recommender systems [1, 18, 29, 32, 34], crowdsourcing and human computation [4,14], and games-with-a-purpose [2,2,6,36,37,39]. We focus in particular on classical recommender systems (such as collaborative filtering and content-based approaches) and games-with-a-purpose to ask whether useful recommender systems data might be crowdsourced as a by-product of casual gameplay. In fact, as we shall see, the idea of crowdsourcing recommendation knowledge is gaining momentum (see, for example, [9, 20, 21, 25]) and we will discuss some specific examples of how these ideas have been adopted by some in the recommender systems community.

ACM ISBN X-XXXXX-XX-X/XX/XX.
DOI: http://dx.doi.org/10.1145/0000000.0000000

2.1 Recommender Systems

Generally speaking there are two common approaches when it comes to building a recommender system. Both rely on the availability of user profiles but each uses different types of information in these profiles and generates recommendations in different ways.

The most well-known recommendation approach is *collaborative filtering* which can be traced back to early work by [30]. User profiles usually take the form of user ratings over some set of items. These ratings may be provided explicitly by the user (e.g. star ratings, likes, etc.) or they may be inferred from user behaviour (e.g. purchase actions, clicks, search histories etc.). Ratings may be unary (e.g. Facebook 'likes'), binary (positive vs negative) or they may be multi-valued (e.g. Amazon's 5-point rating scale). One common collaborative filtering approach is to use these ratings directly to identify users who are similar to the target user and then select highly rated items from their profiles as suggestions for the target user; this approach is known as *user-based* collaborative filtering; see [30]. The same ratings can be used to estimate item similarities (based on ratings correlations) to suggest to the target user movies that are similar to those she has liked; so-called *item-based* collaborative filtering [33]. More recently, researchers have used matrix factorization and related ideas [19] to discover latent features within the ratings data as the basis for recommendation and prediction.

A second common recommendation approach is content-based recommendation; see [29,34]. Unlike collaborative filtering, content-based approaches rely on rich product descriptions; for example, a movie might be described in terms of its genre, actors, director, year or release etc. This data can be used to directly determine similarities between movies. Then, for a target user, recommendations can be produced by, for example, selecting and ranking items that are similar to those that the user has liked in the past.

Collaborative filtering and content-based methods have their pros and cons. The former benefits from large populations of active users, the latter from rich item descriptions. Content-based approaches can comfortably handle new items during recommendations whereas collaborative filtering approaches can only recommend new items once enough ratings have been obtained. Content-based approaches tend generate recommendations that are similar to each other and, as such, can offer limited recommendation diversity. Collaborative filtering are less susceptible to diversity issues but do tend to skew towards more popular items. However both approaches have been used to good effect and can be combined to create hybrid recommenders [5] to offer a *best-of-both-worlds* advantage.

For the purpose of this work we will focus on how a GWAP can be used to collect useful data about which users may be interested in which movies and how this data can be used to generate and rank recommendations, directly and indirectly. In due course, we will also compare these recommendations to those produced by more conventional collaborative filtering and content-based approaches.

2.2 Games-with-a-Purpose

Games-with-a-purpose are motivated by the observation that millions of people enjoy spending time playing games everyday and the tantalising prospect that it may be possible to turn some of this gameplay into solutions (or fragments of solutions) for challenging, large-scale, real-world problems. Many of these problems are classical problems such as image labeling or object recognition but others hint at the power of GWAPs to target some of life's biggest challenges, from drug discovery and protein folding to climate change.

GWAPs often trace their origins to the work of Luis von Ahn [39]. The quintessential GWAP is the ESP Game mentioned in the introduction section of this paper [41]. The ESP Game is an image labelling game; or rather the gameplay date derived from the ESP Game can be used to label images. As already mentioned, it is a two-player game in which two (random) remote players (who do not know each other and cannot communicate) are presented with the same (input) image. The goal of the game is for each player to guess a label (output) the other player will give; this style of game is referred to as an *output agreement* model. Points are awarded, and a new image is presented, when one of the players types a label that matches a label already entered by the other player. Gameplay is enjoyable and addictive, as evidenced by the large number of players and significant investment in gameplay that the ESP Game was able to attract. And as a result of this gameplay it is possible to quickly generate high quality image labels; for example, if a label is frequently entered by players for the same image then it is a strong signal that this label is valid and important for the image. The ESP Game experimented with various features to improve gameplay and encourage the submission of alternative or unusual labels for images as well as popular labels.

Another example of a well-known GWAP is TagATune; see [24]. This time two random players receive inputs (music) that are known by the game, but not the players, to be the same or different. The players provide outputs (tags) describing what the hear so that their partner may be able to assess whether they are listing to the same tune or not. And they gain points if both players correctly determine whether they are listening to the same or a different (input) tune. Thus, this style of GWAP is referred to as an *input agreement* game; see also [23].

Yet a third style of game is exemplified by Peekaboom [43], this time for locating objects in images. It is an example of the so-called *inversion-problem* model of GWAP. In Peekaboom one player ("Peek") attempts to guess the word associated with the image that is being slowly revealed by the other player ("Boom"). Boom can gradually reveal the image in 20-pixel regions and indicate to Peek whether the guesses are "hot" or "cold". If Peek guesses correctly both players receive a score. When this happens the game has generated not just an object label but information about where in the image the labeled object is located, since Boom is motivated to reveal that part of the image that goes with a particular label. The game once again proved popular and playable, attracting large numbers of players and generating significant object label data. Intriguingly, the output of the ESP Game can be used as the image labels to drive Peekaboom.

Over the last few years GWAPs have been proposed for a wide range of tasks, from improving image search [2] to protein folding [6] to large scale urban image acquisition [36]. In this paper we are especially interested in the idea that a GWAP might be a useful way to crowdsource user preferences and other forms of recommendation knowledge. This idea is not new per se, at least in the sense that GWAPs have in the past been used to elicit user preference information. Perhaps the best known example of this is the Matchin game [13] which attempts to learn image preference information by asking two randomly chosen players, "which of these two images do you think your partner prefers?" If both partners click on the same image, they both obtain points, whereas if they click on different images, neither of them receives points; it's another *output agreement* game. The game, although simple, has proven to be enjoyable, attracting tens of thousands of players and gathering millions of preference judgements. The work of [13] compares several techniques for combining these judgments between pairs of images and presents a novel algorithm for recommending unseen images to a target user based on their past judge-

ments. In addition, and as an aside, they go on to show how merely observing user preferences on a specially chosen set of images can accurately predict a user's gender.

2.3 On Crowdsourcing and GWAPs for Recommender Systems

The work of [13] provides an early example of an intriguing link between GWAPs, human computation, crowdsourcing and recommender systems. In the meantime the recommender systems community has grown increasingly aware of, and interested in, such approaches, as evidenced by a series of annual workshops on *Crowdsourcing & Human Computation for Recommender Systems* [1].

For example, the work of [21] considers the sparsity problem in collaborative filtering, by appealing to the crowd as a source of additional information. By using reciprocal recommendations to identify not only items that are suited to users, but also users that are suited to items, they propose that it is possible to create an incentivization for users to contribute information on items. Users are motivated to contribute because the recommender system matches them with items that they find fun and interesting to comment on, review or interact with. The expected result: reduced information sparsity for an overall improvement of recommendations; see also the work of [25] for related ideas.

Felfernig et al. [9] describe how crowdsourcing ideas can be used to address some of the knowledge acquisition and engineering bottlenecks that come with some recommender systems. The authors focus on constraint-based recommenders which harness complex constraint sets that are difficult to acquire. They translate this complex task into a simpler set of micro-tasks (e.g., input an item, or validate an item against a set of characteristics) that are amenable to human computation and crowdsourcing. The crowdsourced data is then automatically converted into a richer set of constraints for the purpose of recommendation and reasoning.

On the matter of GWAPs and recommender systems one notable piece of related work describes the Curator system, a game-with-a-purpose for recommending collections of items that go together [44]. The authors present a class of GWAP for building collections where users create collections; in this case clothing or accessory collections. Players are awarded points based on the collections that match, using an output agreement model. The data from these games helps researchers to develop guidelines for collection recommender systems by, for example, noting items that are frequently collected together versus those similar items that are rarely part of the same collection.

3. THE RECOMMENDATION GAME

Our game is a simple Facebook app based around the idea of a player matching movies with their friends. Facebook was a natural platform choice, not just because of its popularity, but also because it provides access to a user's social graph (specifically the player's friends) and unary preference information (in the form Facebook likes). We will focus on movies here but of course there is no reason why a similar approach could not be used for other types of items or content such as books, music, TV shows, brands, etc. In this section we will summarise the basic architecture and game mechanics, as well as the data produced during gameplay, before detailing how this data can be used in a recommendation context.

3.1 Game Architecture

The overall system architecture is summarised in Figure 1. There are two sides to the system: the *game engine* and the *recommendation engine*. The former is responsible for managing gameplay and collecting relevant (recommendation) data as a side effect of gameplay; we will focus on this in what follows. The latter is responsible for using gameplay data to generate recommendations and we will focus on this aspect in the next section.

The system draws on two external sources of data. As mentioned previously, Facebook provides access to important user data. This includes information about a user's friends ($Friends(u)$) and their avatars, which are needed by the game engine as game targets. It also includes information about *movie likes*, that is movies that a user has explicitly liked on Facebook ($Likes(u)$). The likes of a player's friends represents one source of movies for the game.

The second source of movie data is provided by Rotten Tomatoes[2], a popular movie review site. It is used as source of movie poster graphics to represent the movies during gameplay. But it also used as an additional source of movies for gameplay (popular movies are mixed with the likes of friends during a typical game).

These sources of data, and the data generated by gameplay itself, are used to populate three data-stores of user (friends and likes), movie (posters and popular titles), and game data (matches and interests) as shown.

3.2 Basic Gameplay

Gameplay is designed to be simple but enjoyable. An example screenshot of the game in action is shown in Figure 2. During each game the player p is presented with a set of friend avatars $(f_1, ..., f_n)$ at the bottom of the game arena as shown; currently $n = 5$ friends are selected randomly from $Friends(p)$. In addition a set of movies $(m_1, ..., m_k)$ are chosen from the likes of these friends and including a mixture of popular movies from Rotten Tomatoes; $k = 18$ movies are currently chosen.

The objective of the game is for p to match movies with friends, on the basis that she believes a friend will like a particular movie. Player p does this by dragging and dropping a movie poster onto a friend's avatar. We refer to this as a *match* and use $match(p, m, f)$ to indicate that player p has matched movie m with friend f. The set of matches that make up a given game is denoted by $Matches(p)$ as shown in Equation 1.

$$Matches(p) = \bigcup_{\forall f \in Friends(p)} \{(p, m, f) : match(p, m, f)\} \quad (1)$$

To make the game more challenging, the movie posters follow different trajectories across the screen, becoming more erratic as the game progresses. The player has a limited time to make as many matches as they can. Each match is rewarded with a graphical and audible flourish (the friend's avatar explodes in a fountain of popcorn) and the player receives a variable score. We will discuss scoring shortly but first it is useful to describe the two different forms of recommendation data that can be derived from these matches.

3.3 From Matches to Recommendation Data

For a given match we either know that f likes m ($m \in Likes(f)$), which we refer to as a *known match*, or we have no such knowledge ($m \notin Likes(f)$), in which case it is an *unknown match*. In either case, we can infer useful recommendation data as follows.

In the case of a known match we learn something about p's understanding of f's (movie) interests. Intuitively, if p produces a lot of known matches for some friend f then it suggests that p knows f's interests well because their intuitions are, in some sense, con-

[1] http://crowdrecworkshop.org/

[2] see http://www.rottentomatoes.com

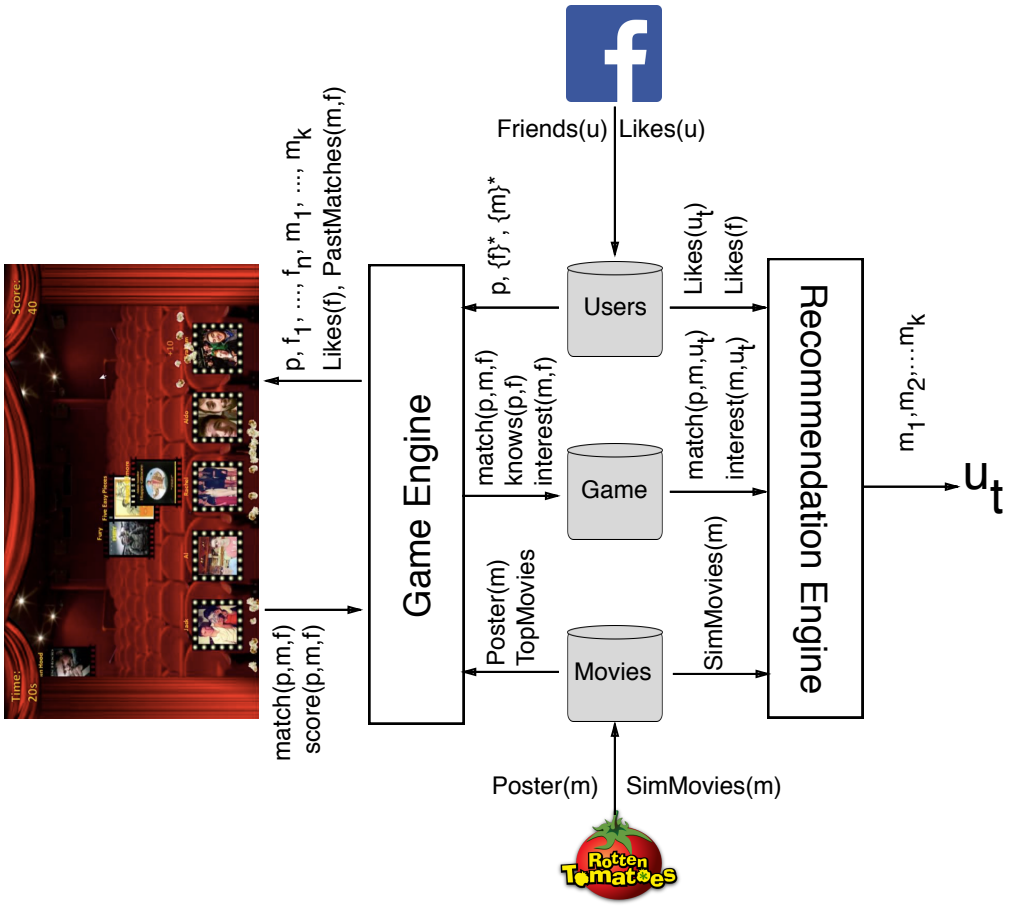

Figure 1: The Recommendation Game architecture, emphasising the game and recommendation components, information flows, and key sources of data, internal and external.

firmed by f's own Facebook likes. We can estimate $knows(p, f)$ as the proportion of known matches ($KnownMatches(p, m, f)$) that p generates for f relative to all matches p generates for f ($FriendMatches(p, m, f)$); see Equations 2 – 4.

$$FriendMatches(p, f) = \{(p, m, f^*) \in Matches(p) : f = f^*\} \quad (2)$$

$$KnownMatches(p, f) = \{(p, m, f) \in FriendMatches(p, f) : m \in Likes(f)\} \quad (3)$$

$$knows(p, f) = \frac{|KnownMatches(p, f)|}{|FriendMatches(p, f)|} \quad (4)$$

As for unknown matches, even when we have no information about f's interest in m ($m \notin Likes(f)$) this does not mean it is a poor match; remember $Likes(f)$ is not exhaustive and so there may be many movies that f likes but that are missing from her Facebook data. The fact that p assigns the match suggests that p thinks f will be interested in m. This establishes m as a potential (novel) recommendation candidate for f in the future. If other players also match m with f then this strengthens the possible relevance of m to f. Equation 5 captures this idea as an interest score based on the number of players who have matched m with f, and how well they know f; their tie strengths with f.

$$interest(m, f) = \sum_{\forall p: match(p,m,f) \wedge m \notin Likes(f)} knows(p, f) \quad (5)$$

Thus, using these two types of matching data (known and unknown matches) we derive relationship information from known matches and we can use this information to weight the plausibility of novel recommendation candidates derived from unknown matches. Later we will describe how this information can be used in a number of different recommendation strategies, but first let us return to the scoring element of gameplay.

3.4 Scoring

Scoring is an important element of game mechanics and, in particular, how a player's score changes provides important feedback and an incentive to play. Scoring is more nuanced that might first appear. We want to encourage the player to make many matches — more matches mean more recommendation data — but we also want to reward useful matches and novel recommendation data. Intuitively we should reward p when she correctly matches m with f; that is when $m \in Likes(f)$. These matches are known to be correct in the sense that m is already known to be liked by f. But what about if m is not known to be liked by f? Such a match can still

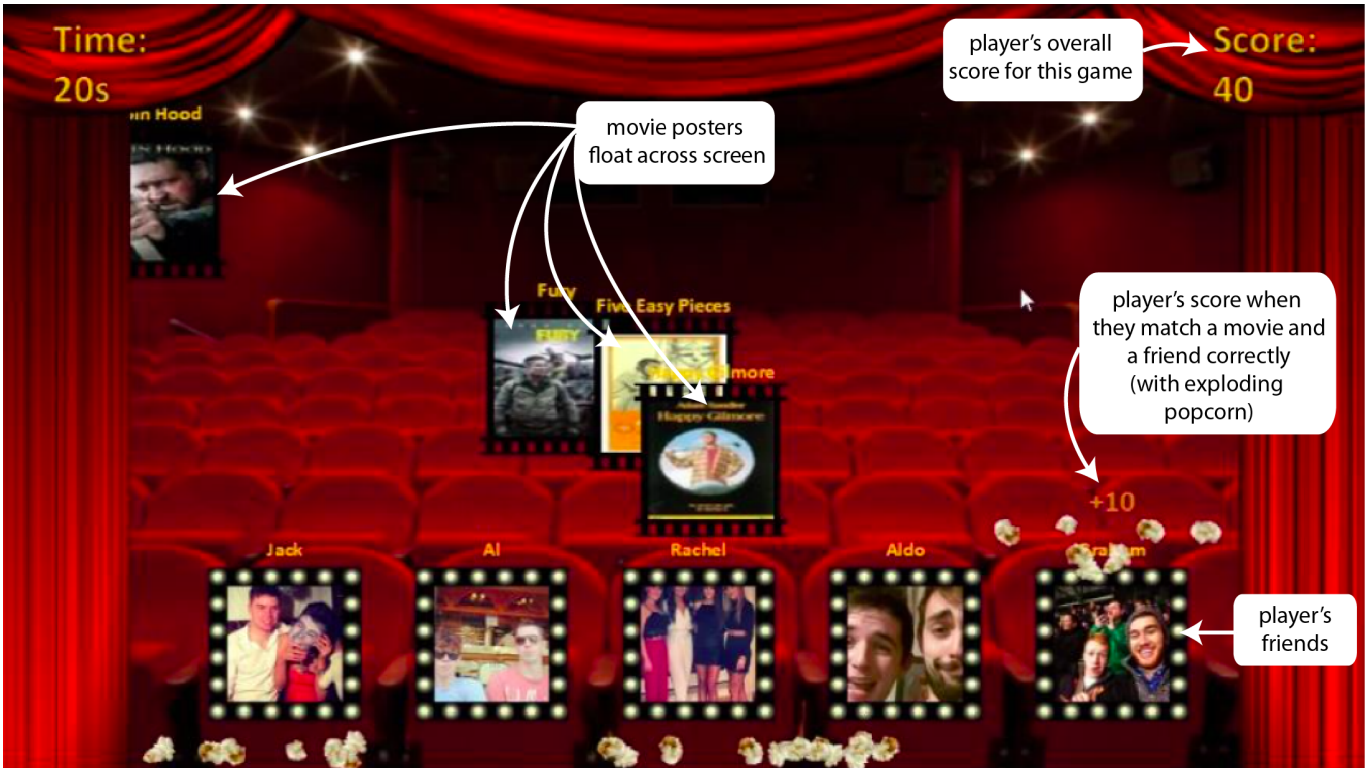

Figure 2: The Recommendation Game in action. The game-area shows the avatars of 5 friends for the current player at the bottom of the screen. Above these we see a number of movie poster graphics, which float across the screen. If a player believes a friend will like a movie then they can drag the movie poster to the corresponding avatar to be rewarded with a graphical flourish (exploding popcorn) and a score. The scoring mechanism is designed to incentivise players to make as many matches as possible while rewarding matches based on an estimate of how useful they are likely to be as recommendation data.

be a useful source of recommendation data because it can suggest a new movie for recommendation to f.

Our scoring metric should give credit for both types of matches, known and unknown. For example, in this work we use Equation 6. In this case α can be used to adjust the relative weight given to known versus unknown matches (in this work, for simplicity, we set $\alpha = 0.5$). The score for unknown matches is inversely proportional to the number of times the same match has been generated by other players in the past ($PastMatches(m, f)$).

Equation 6 returns a score up to 10 for each match. For example, if p matches m with f, such that $m \in Likes(f)$, and 3 other players have made the same match before, then p will receive a score 6.25 ($= 10 \bullet (0.5 \bullet 1 + 0.5/4)$). If, on the other hand, $m \notin Likes(f)$ and p is the first to make such a match then p receives a score of 5 ($= 10 \bullet (0 + 0.5/1)$).

$$score(p,m,f) = 10 \bullet \left(\alpha \bullet 1[m \in Likes(f)] + \frac{1-\alpha}{1 + PastMatches(m,f)} \right) \quad (6)$$

Obviously there are many possible variations on this scoring metric that could (and should) be tested in the future. For example, in the metric above the score given to unknown matches is inversely proportional to how often the same matches have been made in the past; this was decided on the grounds that the most recent match is not producing a new potential recommendation candidate but rather validating an existing one. It could be argued that the unknown match score should be directly proportional to the number of past matches on the grounds that more past matches suggest a higher likelihood that m will turn out to be a good candidate for f. It would be straightforward to implement this but for reasons of space we have left this as a matter for future work.

4. RECOMMENDATIONS STRATEGIES

So far we have described the type of recommendation data that can be produced as a side effect of gameplay, which includes user-user relationship information (tie strength from known matches) and novel recommendation candidates (based on unknown matches). In this section we discuss how these data can be used in an actual recommendation system. In fact we describe 3 different strategies, two using gameplay data in complementary ways, and a third that uses a purely content-based approach for the purpose of comparison. These strategies are, by design, simple and straightforward. They reflect fairly conventional approaches to recommendation rather than the current state of the art in recommender systems. Remember the aim is not to develop new recommendation techniques per se but rather to gain an understanding of the value of the gameplay data as part of such conventional recommendation techniques. If gameplay data proves to be successful in simple recommendation setups then we can be optimistic that it may prove similarly useful in more sophisticated recommendation settings. Given this, the best way to proceed is to start with the simplest possible recommendation strategies. We present each strategy in terms of

how recommendation candidates are selected and then ranked for some target user u_t.

4.1 Crowdsourced Recommendations (CS)

First we adopt an approach that is based purely on crowdsourced (gameplay) data to both generate and rank candidate recommendation items. The candidates are novel movies (that is, unknown to u_t) that were matched with u_t by a player-friend during regular gameplay; see Equation 7. These candidates are movies that u_t should like according to her friends.

$$CS\_Candidates(u_t) = \bigcup_{\forall p \in Friends(u_t)} \{m : match(p, m, u_t) \wedge m \notin Likes(u_t)\} \quad (7)$$

Next, these candidates are ranked by decreasing $interest(m, u_t)$, as per Equation 5. In this way movies that are frequently matched with u_t by many friends who know u_t's tastes well will be ranked higher than movies less often matched with u_t by players less familiar with u_t.

4.2 Collaborative Filtering Recommendations (CF)

Next we implement a version of classical collaborative filtering [18,32] by choosing movies from the profiles (likes) of u_t's friends as recommendation candidates; see Equation 8. The idea here is to select movies liked by friends of u_t but that are not (yet) liked by u_t under the collaborative filtering assumption that friends (similar users) are likely to like many of the same movies.

$$CF\_Candidates(u_t) = \bigcup_{\forall f \in Friends(u_t)} Likes(f) - Likes(u_t) \quad (8)$$

Collaborative filtering typically ranks such recommendation candidates based on the similarity between u_t and the friends or neighbours from where the items originate; movies liked by a more similar user to u_t are more likely to be liked by u_t. User similarity is usually based on some form of ratings correlation in traditional collaborative filtering approaches. Here we instead use the $knows(f, u_t)$ data, which estimates how well f knows u_t, as a proxy for this user-user similarity. We score each candidate recommendation according to the sum of the $knows$ score between its source user and u_t. In other words we again rank candidates in decreasing order of $interest(m, u_t)$. Thus, the CS and the CF techniques differ primarily in the source of the candidates (gameplay vs. profiles, respectively).

4.3 Content-based Recommendations (CB)

Finally, as a convenient and complementary content-based recommendation strategy [29, 34] we use the Rotten Tomatoes API call, $movie\_similar(m)$, which returns a set of 5 movies similar to a given m, based on a variety of meta-data features. In this case we use u_t's likes as seed movies and retrieve the Rotten Tomatoes similar movies for each of these likes. Then the candidate movies for u_t are the concatenation ($\bigoplus$) of all of these Rotten Tomatoes movies returned; see Equation 9.

$$CB\_Candidates(u_t) = \bigoplus_{m \epsilon Likes(u_t)} movie\_similar(m) \quad (9)$$

Content-based recommendations are selected from these candidates at random. Note, $CB\_Candidates(u_t)$ may contain duplicates if the same movies are returned by Rotten Tomatoes for different profile/seed movies. Hence this random selection process will tend to select movies that are more frequently represented in the candidate list. This makes sense as it effectively gives priority to those movies that are considered similar to many movies in u_t's profile thereby reflecting common themes within a user's profile.

This strategy obviously does not make use of any gameplay data. It is included as a useful benchmark for comparison against the previous strategies which both do make use of gameplay data to a greater or lesser degree.

5. EVALUATION

The primary aim of this work is to explore the potential for GWAPs to produce useful recommendation data. To this end the key objective in this section is to evaluate the potential utility of the recommendation data that is produced from our movie matching game. We describe a small pilot study involving 3 different groups of users. The users in each group play the game and then participate in a recommendation test to provide direct feedback on the relevance of a set of movie recommendations based on the above recommendation strategies.

5.1 Setup & Method

Our pilot trial involved 27 participants made up of a mixture of young (approx. 18 - 30 years old) male and females, both undergraduate and postgraduate students at our institution. The participants were made up of 3 groups of 15 (Group 1), 6 (Group 2), and 6 (Group 3). Groups 1 and 2 were sets of close mutual friends; in other words they could be expected to know each other's movie tastes well. In contrast Group 3 was made up of people who did not know each other's movie tastes well, if at all; thus Group 3 served as a useful control when it came to understanding the influence of relationship strength on the recommendation outcome.

Each group of users acted as both players and friends for the purpose of evaluation. In other words, each participant played the recommendation game, participated in the follow-up recommendation test, and their profiles also served as friends in the games of other players. For each participant we had access to their movie likes from Facebook.

The pilot study took place in two phases. Phase 1 focused on collecting recommendation data by asking users to play the game. The data collected in Phase 1 was then used in Phase 2 to generate recommendations for each of the users. We tested our 3 different recommendation strategies and users were asked to rate the resulting recommendations as satisfactory or not.

5.2 Phase 1 - Testing Gameplay

During the first part of the evaluation each user was provided with a link to the game and asked to play it a few times. During each game they were presented with a random subset of 5 friends and 18 movies. These 18 movies were made up of a mixture of movies drawn from the likes of their friends and the most popular movies on Rotten Tomatoes. This ensured a mix of movies as potential known and unknown matches.

On average participants played 11 games each and during these games they matched more than 11 movies per player per game; just over 3 of these matches were known to be correct on average. Table 1 summarises key gameplay statistics on a group by group basis and on average overall (per participant).

In terms of the generation of potential recommendation data this means that, during a typical game, the average player suggested about 8 movies as candidate recommendations; movies that were not already known to be liked by the matched friends. During

Gameplay Stats	G1	G2	G3	Mean
Members/Group	15	6	6	-
Movies/Profile	20	11	13	16.4
Games/Player	15	5	7	11
Matches/Game/Player	10.85	11.03	13.2	11.4
Known/Game/Player	2.76	3.93	3.07	3

Table 1: Gameplay statistics per group and overall per player.

these same games players correctly matched 3 movies per game with friends, which contributed the evaluation of how well a player was likely to know these friends. Thus during a typical gaming session (an average of 11 games per player) this gameplay data quickly builds to provide a detailed source of recommendation data.

5.3 Phase 2 - Evaluating Recommendations

The data collected in Phase 1 was used in Phase 2 to generate recommendations for each of the 27 participants. In each case we generated 6 ranked recommendations from each of the 3 different recommendation strategies and we asked each participant to rate all 18 recommendations as either satisfactory or unsatisfactory; that is, a simple binary rating. When producing these recommendation lists we were careful to randomise the interleaving order of recommendations from each of the strategies to ensure that there was no positional bias.

5.4 Results & Discussion

The results are presented in Figure 3 as a bar chart showing the satisfaction feedback for each of the 3 groups (Groups 1 - 3) of participants and for each of the 3 recommendation techniques (CS, CF, and CB). In each case the percentage satisfaction value is the percentage of recommendations generated by a given algorithm that enjoyed a positive satisfactory rating; so a satisfaction score of 70% for some algorithm means that 70% of the recommendations produced by that algorithm were rated as satisfactory by the participants.

We can see that the CF and CB strategies tend to produce similar satisfaction scores of between 70% - 80% across the 3 groups. However a very different result is evident for the CS strategy, which relies wholly on gameplay data. In this case we can see that the recommendations produced for Group 1 and 2 participants (those with strong social connections) enjoy much higher satisfaction scores, in both cases approximately 94%. Interestingly we see that this increased satisfaction level is not maintained by CS for Group 3. The Group 3 satisfaction scores average only 76% for CS, comparable with those for CF and CB, most likely because, unlike Groups 1 and 2, Group 3 participants are not close friends and so do not know each others movie tastes well. As a result Group 3 users should find it more difficult generate good matches for their 'friends' in the game thereby limiting the recommendation data that is produced as a result.

It is also interesting to examine the diversity of recommendations that are being made by the different approaches; diversity is often considered to be a useful measure in modern recommender systems as it speaks to the ability of the recommender to offer the user more or less choice through its suggestions [15, 35, 38]; all other things being equal, more diversity is generally viewed as desirable. As a simple measure of diversity we can look at the percentage of unique recommendations made by each of the 3 approaches and the results are presented in Figure 4.

This time the CB strategy performs best, producing recommendation lists with much higher diversity than both the CS and CF approaches. For example the average diversity of CB recommendations across the 3 groups is 77% (that is 77% of the recommendations are unique) compared to only 34% and 43% for CS and CF respectively. This is likely due to the limited size of user profiles in this pilot, which ultimately constrains the set of available movies for gameplay and recommendation. By comparison the CB approach calls on a much larger set of movies available through Rotten Tomatoes.

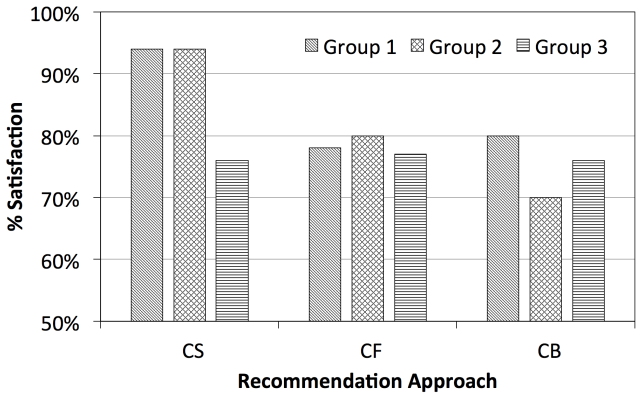

Figure 3: Recommendation satisfaction scores by group and by recommendation strategy.

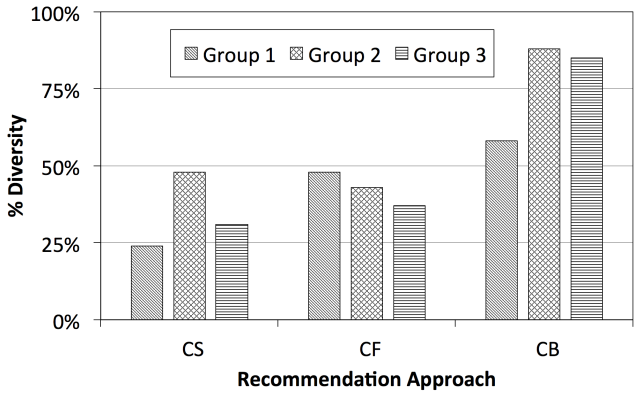

Figure 4: Recommendation diversity scores by group and by recommendation strategy.

Another part of the explanation for the low diversity scores for CS (compared with CF) is likely to be that gameplay data skews towards popular movies. Players recognise the icons/posters of popular more readily and naturally gravitate towards these during gameplay. As a result players will tend to focus on a more limited set of movies, hence the lower diversity score at recommendation time. Moreover, because these movies are popular they may also be easier to match with friends and there will be a greater likelihood that they will be liked, hence the improved satisfaction scores.

It is a matter for future work to consider this relationship between satisfaction and diversity further. It may be possible to guide gameplay towards more novel items by manipulating the size and speed of items according to the item popularity. For example, perhaps unusual movies could be presented with a larger icon and/or

a slower trajectory, making them more attractive gameplay targets; scoring could also be adapted with respect to inverse item popularity to incentivize this type of behaviour.

6. DISCUSSION

The recommendation game, as described, is of course just one example of how we might use a GWAP to generate the type of data that can be useful in recommender systems. It is certainly interesting to consider alternative game styles as additional opportunities to generate useful recommendation data, perhaps for different types of recommendation tasks.

A similar type of matching game might be considered to generate the type of data that a group recommender systems might find useful; see [16]. Normally, group recommender systems assume a fixed group and the task is to identify items that will satisfy the group as a whole. An alternative formulation might be to start with an item and identify a group of users who will like the item. Indeed this format might be better adapted to group recommendation tasks such as matching movies with friends because rather than make compromises by selecting a movie for a fixed set of friends it may be better to identify a suitable set of friends for a given movie. With this in mind one could imagine a similar matching game to that presented in this paper but where friends are floating across the game arena and the task of the player is to match these friends with one of a fixed set of movies currently on release. In this way gameplay will associate groups of friends with movies. And if there is a tendency for players to associate the same (or similar) groups of friends with a movie then we can gain valuable information about the type of groups that are likely to enjoy that movie.

Another game mechanic can be borrowed from simple "snap" style games. For example, it is easy to envisage a two-player game in which each player is presented with the same friend-movie pairing. The task of each player is to quickly indicate whether the pairing is good or bad (indicating that the friend will like the movie or not, respectively). If both players agree then they both receive a score (perhaps with the fastest player receiving a bonus). If they disagree they get no score. In this way, player-agreement determines which pairings are likely to be good ones. This same game mechanic can be used to validate movie-movie pairings to obtain item-item similarity data for example.

These are just two simple examples of further GWAPs that we intend to evaluate. The trick will be to find compelling game mechanics that can be used as the basis for collecting useful recommendation data at scale. If the game is fun and addicting then it has the potential to attract large numbers of users and generate huge amounts of valuable recommendation data. Even in our small scale user study it was interesting to see that participants appeared willing to play multiple games with each game generating multiple pieces of recommendation data. For example on average each participant played an average of just over 11 games and generated about 8 new recommendation candidates. That makes for a total of 88 new recommendation candidates generated per participant. This level of recommendation data would be challenging to solicit using more traditional survey-based techniques.

7. CONCLUSIONS

The main purpose of this paper is to explore the use of a GWAP to collect recommendation data and user preferences as a side effect of casual gameplay. This is interesting because, if successful, this crowdsourcing approach has the potential to deliver preference data at scale; there are many examples of GWAPs that attract large communities of highly engaged users [42]. We have implemented and tested a simple movie matching game and described the recommendation data that we can collect from its gameplay. These data can be used in a recommendation context and we have demonstrated its potential as part of a live-user trial.

Obviously the work in this paper is limited in many ways. The developed system is a working prototype and the small scale of our evaluation is just a first step to explore the potential role of GWAPs in recommender systems. Nevertheless we believe it serves to highlight the potential for new ways to think about recommendation data and recommender systems while contributing to a growing interest in the role of crowdsourcing in recommender systems and personalization research.

Our attention in this paper has been on the movie domain, a classical recommendation domain. Of course the approach is not limited to this specific domain, and it is likely that similar techniques can be used for other types of content such as music, TV shows, brands etc. In addition, our focus has been on deriving specific types of recommendation data, preference data and user similarity data. It will be interesting to consider whether related approaches can be used to source other forms of recommendation data such as item similarities [33], trust or reputation data [11, 26, 27], or even opinion sentiment [8] and explanations [2].

Obviously we have focused on a simple, single-player matching game and there are a great many opportunities to explore more sophisticated game mechanics. For example, our current work is exploring multi-player game designs for group recommendation tasks. Adding additional players provides an opportunity to further scale-up the generation of useful data via multi-party validation. It also introduces new game dynamics including strategic gameplay elements that have the potential to provide for a more long-lasting gaming experience.

8. ACKNOWLEDGEMENTS

This work is supported by Science Foundation Ireland under grant 07/CE/I1147.

9. REFERENCES

[1] Adomavicius, G., and Tuzhilin, A. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering 17*, 6 (June 2005), 734–749.

[2] Ahn, L. V., Ginosar, S., Kedia, M., and Blum, M. Improving Image Search with PHETCH. *Proceedings of the 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07 4* (Apr. 2007).

[3] Banks, S., Rafter, R., and Smyth, B. The recommendation game: Using a game-with-a-purpose to generate recommendation data. In *Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16-20, 2015* (2015), 305–308.

[4] Bigham, J. P., and Parkes, D. C., Eds. *Proceedings of the Second AAAI Conference on Human Computation and Crowdsourcing, HCOMP 2014, November 2-4, 2014, Pittsburgh, Pennsylvania, USA*, AAAI (2014).

[5] Burke, R. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction 12*, 4 (Nov. 2002), 331–370.

[6] Cooper, S., Khatib, F., Treuille, A., Barbero, J., Lee, J., Beenen, M., Leaver-Fay, A., Baker, D., Popović, Z., and Players, F. Predicting Protein Structures with a Multiplayer Online Game. *Nature 466*, 7307 (2010), 756–760.

[7] Desrosiers, C., and Karypis, G. A Comprehensive Survey of Neighborhood-based Recommendation Methods. In *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. 2011, 107–144.

[8] Dong, R., Schaal, M., O'Mahony, M. P., and Smyth, B. Topic extraction from online reviews for classification and recommendation. In *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013* (2013).

[9] Felfernig, A., Haas, S., Ninaus, G., Schwarz, M., Ulz, T., Stettinger, M., Isak, K., Jeran, M., and Reiterer, S. RecTurk: Constraint-based Recommendation based on Human Computation. In *Proceedings of the 3rd International Workshop on Crowdsourcing and Human Computation for Recommender Systems at ACM RecSys - CrowdRec'14* (Foster City, CA, USA, Oct. 2014), 1–6.

[10] Gligorov, R., Hildebrand, M., van Ossenbruggen, J., Schreiber, G., and Aroyo, L. On the Role of User-generated Metadata in Audio Visual Collections. In *Proceedings of the 6th International Conference on Knowledge Capture - K-CAP '11*, M. A. Musen and O. Corcho, Eds., ACM (Banff, Canada, June 2011), 145–151.

[11] Golbeck, J. Generating predictive movie recommendations from trust in social networks. In *Proceedings of the 4th International Conference on Trust Management - iTrust'06*, K. Stølen, W. H. Winsborough, F. Martinelli, and F. Massacci, Eds., vol. 3986, Springer Berlin Heidelberg (Pisa, Italy, May 2006), 93–104.

[12] Guo, G., Zhang, J., and Yorke-Smith, N. Trustsvd: Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.* (2015), 123–129.

[13] Hacker, S., and von Ahn, L. Matchin: Eliciting User Preferences with an Online Game. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems - CHI 09*, ACM Press (Boston, Massachusetts, USA, Apr. 2009), 1207.

[14] Hartman, B., and Horvitz, E., Eds. *Proceedings of the First AAAI Conference on Human Computation and Crowdsourcing, HCOMP 2013, November 7-9, 2013, Palm Springs, CA, USA*, AAAI (2013).

[15] Hurley, N. J. Personalised ranking with diversity. In *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013* (2013), 379–382.

[16] Jameson, A., and Smyth, B. Recommendation to groups. In *The Adaptive Web, Methods and Strategies of Web Personalization* (2007), 596–627.

[17] Kelly, D., and Teevan, J. Implicit Feedback for Inferring User Preference: A Bibliography. *ACM SIGIR Forum 37*, 2 (2003), 18 – 28.

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

[19] Koren, Y., Bell, R., and Volinsky, C. Matrix Factorization Techniques for Recommender Systems. *Computer 42*, 8 (Aug. 2009), 30–37.

[20] Larson, M., Cremonesi, P., and Karatzoglou, A. Overview of ACM recsys crowdrec 2014 workshop: crowdsourcing and human computation for recommender systems. In *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 - 10, 2014* (2014), 381–382.

[21] Larson, M., Said, A., Shi, Y., Cremonesi, P., Tikk, D., Baltrunas, L., Karatzoglou, A., Geurts, J., Anguera, X., and Hopfgartner, F. Activating the Crowd: Exploiting User-Item Reciprocity for Recommendation. In *Proceedings of the 2nd International Workshop on Crowdsourcing and Human Computation for Recommender Systems at ACM RecSys - CrowdRec'13* (Hong Kong, 2013).

[22] Law, E., and von Ahn, L. Input-Agreement: A New Mechanism for Collecting Data using Human Computation Games. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems - CHI 09*, ACM Press (Boston, Massachusetts, USA, Apr. 2009), 1197–1206.

[23] Law, E., and von Ahn, L. Input-agreement: a new mechanism for collecting data using human computation games. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems, CHI 2009, Boston, MA, USA, April 4-9, 2009* (2009), 1197–1206.

[24] Law, E. L. M., von Ahn, L., Dannenberg, R. B., and Crawford, M. Tagatune: A game for music and sound annotation. In *Proceedings of the 8th International Conference on Music Information Retrieval, ISMIR 2007, Vienna, Austria, September 23-27, 2007* (2007), 361–364.

[25] Lee, J., Jang, M., Lee, D., Hwang, W.-S., Hong, J., and Sang-Wook, K. Alleviating the Sparsity in Collaborative Filtering using Crowdsourcing. In *Proceedings of the 2nd International Workshop on Crowdsourcing and Human Computation for Recommender Systems at ACM RecSys - CrowdRec'13* (Hong Kong, Oct. 2013).

[26] McNally, K., O'Mahony, M. P., and Smyth, B. A comparative study of collaboration-based reputation models for social recommender systems. *User Model. User-Adapt. Interact. 24*, 3 (2014), 219–260.

[27] O'Donovan, J., and Smyth, B. Trust in recommender systems. In *Proceedings of the 2005 International Conference on Intelligent User Interfaces, January 10-13, 2005, San Diego, California, USA* (2005), 167–174.

[28] O'Donovan, J., and Smyth, B. Is trust robust?: an analysis of trust-based recommendation. In *Proceedings of the 2006 International Conference on Intelligent User Interfaces, January 29 - February 1, 2006, Sydney, Australia* (2006), 101–108.

[29] Pazzani, M. J., and Billsus, D. Content-based recommendation systems. In *The Adaptive Web, Methods and Strategies of Web Personalization* (2007), 325–341.

[30] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. GroupLens: An Open Architecture For Collaborative Filtering Of Netnews. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work - CSCW 94*, ACM (Chapel Hill, North Carolina, USA, Aug. 1994), 175 – 186.

[31] Salvador, A., Carlier, A., Giro-i Nieto, X., Marques, O., and Charvillat, V. Crowdsourced Object Segmentation with a Game. In *Proceedings of the 2nd ACM International Workshop on Crowdsourcing for Multimedia - CrowdMM '13*, ACM (Barcelona, Spain, Oct. 2013), 15–20.

[32] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001* (2001), 285–295.

[33] Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. Item-Based Collaborative Filtering Recommendation Algorithms. In *Proceedings of the 10th International World Wide Web Conference - WWW 10*, ACM (Hong Kong, May 2001), 285 – 295.

[34] Smyth, B. Case-based recommendation. In *The Adaptive Web, Methods and Strategies of Web Personalization* (2007), 342–376.

[35] Smyth, B., and McClave, P. Similarity vs. diversity. In *Case-Based Reasoning Research and Development, 4th International Conference on Case-Based Reasoning, ICCBR 2001, Vancouver, BC, Canada, July 30 - August 2, 2001, Proceedings* (2001), 347–361.

[36] Tuite, K., Snavely, N., Hsiao, D.-y., Tabing, N., and Popovic, Z. PhotoCity: Training Experts at Large-scale Image Acquisition through a Competitive Game. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems - CHI '11*, ACM Press (Vancouver, Canada, May 2011), 1383–1392.

[37] van Zwol, R., Garcia, L., Ramirez, G., Sigurbjornsson, B., and Labad, M. Video Tag Game. In *Proceedings of the 17th International World Wide Web Conference (WWW Developer Track)*, H. Jinpeng, R. Chen, H.-W. Hon, Y. Liu, W.-Y. Ma, A. Tomkins, and X. Zhang, Eds., ACM Press (Beijing, China, Apr. 2008).

[38] Vargas, S., and Castells, P. Improving sales diversity by recommending users to items. In *Eighth ACM Conference on Recommender Systems, RecSys '14, Foster City, Silicon Valley, CA, USA - October 06 - 10, 2014* (2014), 145–152.

[39] von Ahn, L. Games with a Purpose. *Computer 39*, 6 (June 2006), 92–94.

[40] von Ahn, L., and Dabbish, L. Labeling Images with a Computer Game. In *Proceedings of The ACM Conference on Human Factors in Computing Systems*, E. Dykstra-Erickson and M. Tscheligi, Eds., ACM (Vienna, Austria, Apr. 2004), 319 – 326.

[41] von Ahn, L., and Dabbish, L. ESP: labeling images with a computer game. In *Knowledge Collection from Volunteer Contributors, Papers from the 2005 AAAI Spring Symposium, Technical Report SS-05-03, Stanford, California, USA, March 21-23, 2005* (2005), 91–98.

[42] von Ahn, L., and Dabbish, L. Designing Games with a Purpose. *Communications of the ACM 51*, 8 (2008), 57.

[43] Von Ahn, L., Liu, R., and Blum, M. Peekaboom: a Game for Locating Objects in Images. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '06*, R. Grinter, T. Rodden, P. Aoki, E. Cutrell, R. Jeffries, and G. Olson, Eds. (Montréal, Canada, Apr. 2006), 55–64.

[44] Walsh, G., and Golbeck, J. Curator: a Game with a Purpose for Collection Recommendation. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems - CHI '10*, ACM Press (Atlanta, Georgia, USA, Apr. 2010), 2079–2082.