Proceedings of the Tenth International AAAI Conference on
Web and Social Media (ICWSM 2016)

CrowdLens: Experimenting with Crowd-Powered Recommendation and Explanation

Shuo Chang, F. Maxwell Harper, Lingfei He and **Loren G. Terveen**
GroupLens Research
University of Minnesota
{schang, harper, lingfei, terveen}@cs.umn.edu

Abstract

Recommender systems face several challenges, e.g., recommending novel and diverse items and generating helpful explanations. Where algorithms struggle, people may excel. We therefore designed *CrowdLens* to explore different workflows for incorporating people into the recommendation process. We did an online experiment, finding that: compared to a state-of-the-art algorithm, crowdsourcing workflows produced more diverse and novel recommendations favored by human judges; some crowdworkers produced high-quality explanations for their recommendations, and we created an accurate model for identifying high-quality explanations; volunteers from an online community generally performed better than paid crowdworkers, but appropriate algorithmic support erased this gap. We conclude by reflecting on lessons of our work for those considering a crowdsourcing approach and identifying several fundamental issues for future work.

Introduction

Recommendation algorithms are widely used in online systems to customize the delivery of information to match user preferences. For example, Facebook filters its news feed, the Google Play Store suggests games and apps to try, and YouTube suggests videos to watch.

Despite their popularity, algorithms face challenges. First, while algorithms excel at predicting whether a user will like a single item, for sets of recommendations to please users, they must balance factors like the *diversity*, *popularity*, and *recency* of the items in the set (McNee, Riedl, and Konstan 2006). Achieving this balance is an open problem. Second, users like to know *why* items are being recommended. Systems do commonly provide *explanations* (Herlocker, Konstan, and Riedl 2000), but they are template-based and simplistic.

People may not be as good at algorithms at predicting how much someone will like an item (Krishnan et al. 2008). But "accuracy is not enough" (McNee, Riedl, and Konstan 2006). We conjecture that people's creativity, ability to consider and balance multiple criteria, and ability to explain recommendations make them well suited to take on a role in the recommendation process. Indeed, many sites rely on users to produce recommendations and explain the reasons for their recommendations. For example, user-curated book lists in Goodreads[1] are a popular way for users to find new books to read.

However, sites – especially new or small ones – may not always be able to rely on users to produce recommendations. In such cases, *paid crowdsourcing* is an alternative, though paid crowd workers may not have domain expertise and knowledge of users' preferences.

All things considered, while incorporating crowdsourcing into the recommendation process is promising, how best to do so is an open challenge. This work takes on that challenge by exploring three research questions:

RQ1 - Organizing crowd recommendation *What roles should people play in the recommendation process? How can they complement recommender algorithms?* Prior work has shown that crowdsourcing benefits from an iterative workflow (e.g., (Little et al. 2009; Bernstein et al. 2010; Kittur et al. 2011)). In our context, we might iteratively generate recommendations by having one group *generates examples* and another group *synthesizes recommendations*, incorporating those examples along with their own fresh ideas. Alternatively, we might substitute a recommendation algorithm as the source of the examples. We are interested in the effectiveness of these designs compared with an algorithmic baseline. We speculate that this design decision will have a measurable impact on the quality, diversity, popularity, and recency of the resulting recommendations.

RQ2 - Explanations *Can crowds produce useful explanations for recommendations? What makes a good explanation?* Algorithm can only generate template-based explanations, such as "we recommend X because it is similar to Y". People, on the other hand, have the potential to explain recommendations in varied and creative ways. We will explore the feasibility of crowdsourcing explanations, identify characteristics of high-quality explanations, and develop mechanisms to identify and select the best explanations to present to users.

RQ3 - Volunteers vs. crowdworkers *How do recommendations and explanations produced by volunteers compare to those produced by paid crowdworkers?* Volunteers from a site have more domain knowledge and commitment to the site, while crowdworkers (e.g. from Amazon Mechan-

[1] www.goodreads.com

ical Turk) are quicker to recruit. We thus will compare result quality and timeliness for the two groups.

To address these questions, we built *CrowdLens*, a crowdsourcing framework and system to produce movie recommendations and explanations. We implemented CrowdLens on top of MovieLens[2], a movie recommendation web site with thousands of active monthly users.

Contributions. This paper offers several research contributions to a novel problem: how to incorporate crowd work into the recommendation process. We developed an iterative crowd workflow framework. We experimentally evaluated several different workflows, comparing them to a state of the art recommendation algorithm; crowd workflows produced comparable quality, more diverse, and less common recommendations. Crowdworkers were capable of producing good explanations; we developed a model that identified features associated with explanation quality and that can predict good explanations with high accuracy. Volunteers generally performed better than paid crowdworkers, but providing paid crowdworkers good examples reduced this gap.

Related work

Foundational crowdsourcing work showed that tasks could be decomposed into independent microtasks and that people's outputs on microtasks could be aggregated to produce high quality results. After the original explorations, researchers began to study the use of crowdsourcing for more intellectually complex and creative tasks. Bernstein et al. (Bernstein et al. 2010) created Soylent, a Microsoft Word plugin that uses the crowd to help edit documents. Lasecki et al. (Lasecki et al. 2013) used crowdworkers to collaboratively act as conversational assistants. Nebeling et al. (Nebeling, Speicher, and Norrie 2013) proposed CrowdAdapt, a system that allowed the crowd to design and evaluate adaptive website interfaces. When applying crowdsourcing for these more complicated tasks, various issues arose.

Organizing the workflow. More complicated tasks tend to require more complex worklows, i.e., ways to organize crowd effort to ensure efficiency and good results. Early examples included: Little et al. (Little et al. 2009) proposed an iterative crowdsourcing process with solicitation, improvement and voting; and Kittur et al. (Kittur et al. 2011) applied the Map Reduce pattern to organize crowd work. For Soylent, Bernstein et al. (Bernstein et al. 2010) developed the *find-fix-verify* pattern: some workers would *find* issues, others would propose *fixes*, and still others would *verify* the fixes. Similar to this research, we developed an iterative workflow for incorporating crowds into recommendation: *generate* examples, then *synthesize* recommendations.

Types of crowdworkers. Depending on the difficulty and knowledge requirements of a task, different types of crowd workers may be more appropriate. Zhang et al. (Zhang et al. 2012) used paid crowdworkers from Amazon Mechanical Turk to collaboratively plan itineraries for tourists, finding that Turkers were able to perform well. Xu et al. (Xu, Huang, and Bailey 2014; Xu et al. 2015) compared structured feedback from Turkers on graphic designs with freeform feedback from design experts. They found that Turkers could reach consensus with experts on design guidelines. On the other hand, when Retelny et al. (Retelny et al. 2014) explored complex and interdependent applications in engineering, they used a "flash team" of paid *experts*. Using a system called Foundry to coordinate, they reduced the time required for filming animation by half compared to using traditional self-managed teams. Mindful of these results, we compared performance of paid crowdworkers to "experts" (members of the MovieLens film recommendation site).

Crowdsourcing for personalization. *Personalization* – customizing the presentation of information to meet an individual's preferences – requires domain knowledge as well as knowledge of individual preferences. This complex task has received relatively little attention from a crowdsourcing perspective. We know of two efforts that focused on a related but distinct task: *predicting* user ratings of items. Krishnan et al. (Krishnan et al. 2008) compared human predictions to those of a collaborative filtering recommender algorithm. They found that most humans are worse than the algorithm while a few were more accurate, and people relied more on item content and demographic information to make predictions. Organisciak et al. (Organisciak et al. 2014) proposed a crowdsourcing system to predict ratings on items for requesters. They compared a collaborative filtering approach (predicting based on ratings from crowdworkers who share similar preferences) with a crowd prediction approach (crowdworkers predicting ratings based on the requester's past ratings). They found that the former scales up to many workers and generates a reusable dataset while the latter works in areas where tastes of requesters can be easily communicated with fewer workers.

Generating sets of recommendations is more difficult than predicting ratings: as we noted, good recommendation sets balance factors such as diversity, popularity, familiarity, and recency. We know of two efforts that used crowdsourcing to generate recommendations: Felfernig et al. deployed a crowd-based recommender system prototype (Felfernig et al. 2014), finding evidence that users were satisfied with the interface. The StitchFix[3] commercial website combines algorithmic recommendations with crowd wisdom to provide its members with a personalized clothing style guide. Our research also explores incorporating people into the recommendation process. We experimented with several roles for people and evaluated their performance using a range of recommendation quality measures.

The CrowdLens Framework

Though individuals are not as accurate as algorithms in predicting ratings (Krishnan et al. 2008), crowdsourcing gains power by aggregating inputs from multiple people. In this section, we describe several key aspects of CrowdLens, including the intended use of the recommendations, the recommendation workflow, and the user interface.

[2] http://movielens.org

[3] www.stitchfix.com

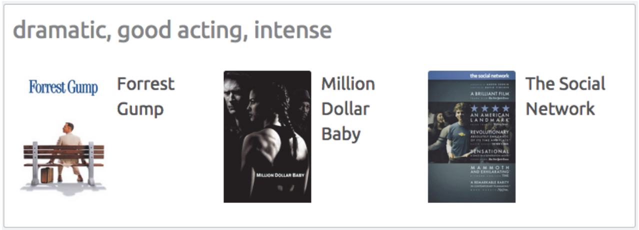


Figure 1: An example movie group. New users in MovieLens express their taste on movie groups.

Recommendation Context: Movie Groups

Human recommendations do not scale like algorithmic recommendations. It is simply too much work to ask users to come up with long lists of personalized content for *all* users of a system. Therefore, human-built recommendation lists are typically non-personalized or semi-personalized: a video game web site lists the top games of the year; a DJ plays music suited for listeners that like a particular kind of music.

In this research, we explore the idea of using humans to generate recommendations for users who have expressed a preference for a particular type of movie. MovieLens uses "movie groups" during the sign-up process: after users register, they are asked to distribute three "like" points across six movie groups, where each movie group is summarized by three representative movies and three descriptive terms. See figure 1 for an example. MovieLens then uses a semi-personalized modification of collaborative filtering to recommend for these new users. First, MovieLens finds existing active users with similar movie tastes and represents a new user's preferences as the average rating vector of these active users. Then the standard MovieLens (item-item) algorithm can be applied. Finally, once a user has rated enough movies him or herself, the system switches to using these actual ratings to represent the user's preference. (See (Chang, Harper, and Terveen 2015) for details.) This algorithm will serve as the baseline of comparison for our crowd recommendation approaches.

We therefore design CrowdLens to collect recommendations and explanations for each of the six movie groups offered to new users of MovieLens.

Crowd Recommendation Workflow

There are many possible designs for generating recommendations from a crowd of workers. Recent research provides evidence that providing good "example" responses can improve the quality (Kulkarni, Dow, and Klemmer 2014) and diversity (Siangliulue et al. 2015) of responses in crowdsourced creative work; however, providing examples also may lead to conformity and reduced diversity (Smith, Ward, and Schumacher 1993). Therefore, we thought it was both promising and necessary to experiment with a recommendation workflow that used examples. CrowdLens organizes recommendations into a two step "pipeline": the first step *generates* a candidate set of recommendations, and a second step *synthesizes* the final set of recommendations, either drawing from the generated candidates, or adding new content. This process may enable both creativity – the first group will come up with more diverse and surprising recommendations – and quality – the second group will gravitate toward and be guided by the best recommendations from the first. This workflow is similar to workflows that have been used successfully elsewhere for crowdsourcing subjective and creative tasks (Little et al. 2009; Bernstein et al. 2010; Kittur et al. 2011).

The first step (generate candidates) in the CrowdLens pipeline can be fulfilled either by a recommendation algorithm or by crowdworkers. This allows us to experiment with different configurations of human-only and algorithm-assisted workflows.

User Interface

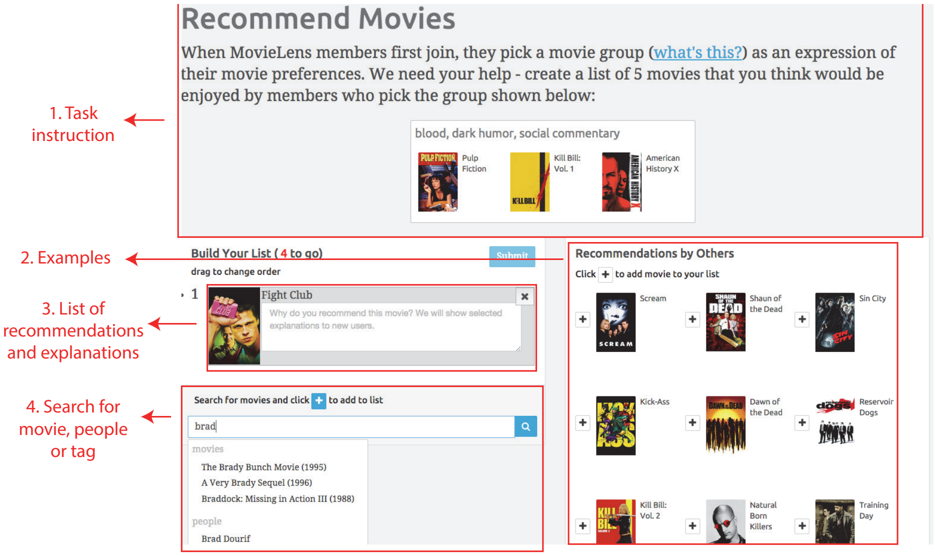
See figure 2 for a screenshot of the CrowdLens recommendation interface. The recommendation task is framed by asking workers to produce a list of **5 movies** that "would be enjoyed by members who pick [a specific movie group]". As detailed in the figure, the interface has four major parts:

1. Instructions for the crowdworker and a description of the target movie group.
2. A set of example recommendations. Crowdworkers may add recommendations from this list by clicking the "+" sign next to each example movie. This component is visible only for workers in the second (*synthesize*) step, not the first (*generate*).
3. The list of movies the crowdworker recommends. Each recommendation is accompanied by a text box where the crowdworker must write an explanation for why they are recommending the movie.
4. An auto-complete search interface, which crowdworkers can use to find movies to recommend. Users can search for movies by title, actor, director, or tags.

Experiment

We conducted an online experiment of crowd recommendation in MovieLens. We recruited participants from MovieLens and Amazon Mechanical Turk to generate and synthesize recommendations and to produce explanations. We evaluated the resulting recommendations and explanations using both offline data analysis and human judgments, which we gathered from another set of MovieLens users.

Participants We recruited 90 MovieLens participants via email invitations between Dec 16, 2014 and Jan 30, 2015. The qualification criterion for MovieLens users was logging in at least once after November 1, 2014. We recruited 90 Amazon Mechanical Turk workers on Jan 30, 2015. Each Turker was allowed to complete one HIT; this ensured that no single Turker was assigned to multiple experimental conditions. We recruited turkers from the US and Canada with approval rate of over 95%, paying them above effective minimum salary in US.



Recommend Movies

When MovieLens members first join, they pick a movie group (what's this?) as an expression of their movie preferences. We need your help - create a list of 5 movies that you think would be enjoyed by members who pick the group shown below:

blood, dark humor, social commentary

Build Your List (4 to go)

drag to change order

1 Fight Club

Why do you recommend this movie? We will show selected explanations to new users.

Search for movies and click + to add to list

brad

movies

The Brady Bunch Movie (1995)

A Very Brady Sequel (1996)

Braddock: Missing in Action III (1988)

people

Brad Dourif

Recommendations by Others

Click + to add movie to your list

Figure 2: CrowdLens user interface. The four core components are task instructions, example recommendations, multi-category search, and the worker-generated recommendation list. The interface for generating examples is the same except without component 2 - examples.

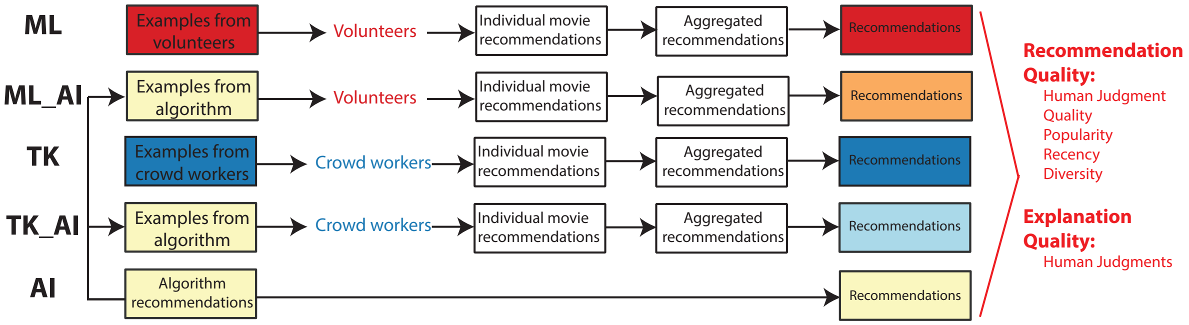


Figure 3: Five pipelines for producing recommendations. The human-only and algorithm-assisted pipelines were instantiated with both MovieLens volunteers and paid crowdworkers from Mechanical Turk. The MovieLens recommendation algorithm served as a baseline and was the source of input examples for the algorithm-assisted pipelines.

Experimental Design We have a 2 × 2 between-subjects design with the following factors:

1. **Type of worker:** *volunteer* (MovieLens user) or *paid crowdworker* (from Amazon Mechanical Turk).
2. **Source of examples:** are example recommendations generated by an *algorithm* or aggregated from another set of *workers*?

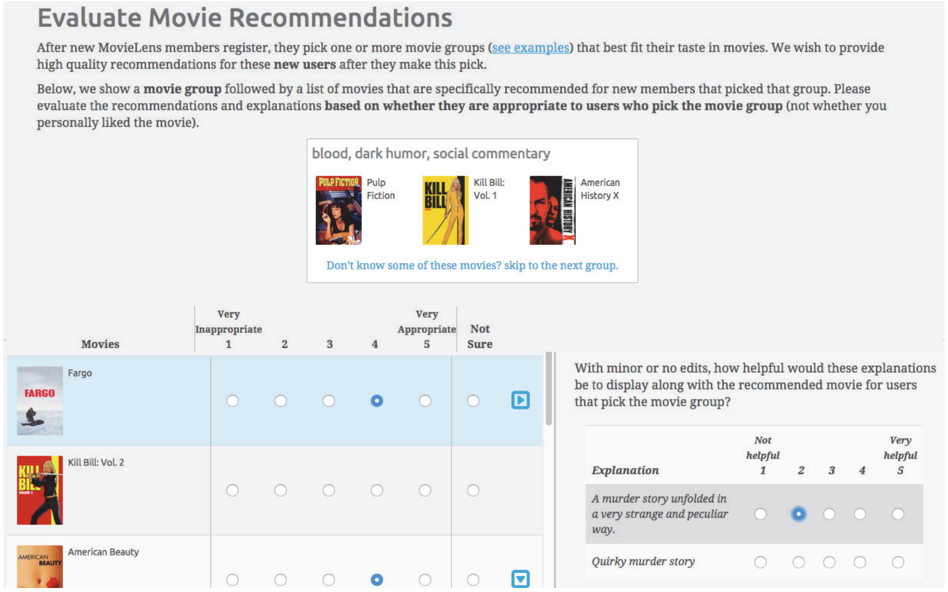
In discussing our results, we refer to the four resulting experimental conditions as **ML**, **ML_AI**, **TK**, and **TK_AI** – see figure 3. These pipelines generate both recommendations and explanations. We also include a baseline condition – the semi-personalized collaborative filtering algorithm (Chang, Harper, and Terveen 2015) described above. We refer to the baseline as **AI**. The baseline algorithm only generates recommendations, therefore there is no baseline condition for studying explanations.

In each of the four experimental pipelines, five participants independently produced a set of 5 recommended movies for each of the 6 MovieLens movie groups. In the human-only pipelines, (**TK** and **ML**), 5 participants generated the initial (example) recommendations. This accounted for the total of 180 experimental participants:

- 90 MovieLens volunteers: (10 (**ML**) + 5 (**ML_AI**)) × 6 (MovieLens movie groups) *plus*
- 90 Turkers: (10 (**TK**) + 5 (**TK_AI**)) × 6 (MovieLens movie groups)

We clarify the recommendation pipelines by walking through an example - recommending for movie group shown in figure 2 in **ML** pipeline.

1. 5 MovieLens volunteers independently recommend movies for the movie group with no examples provided



Evaluate Movie Recommendations

After new MovieLens members register, they pick one or more movie groups (see examples) that best fit their taste in movies. We wish to provide high quality recommendations for these **new users** after they make this pick.

Below, we show a **movie group** followed by a list of movies that are specifically recommended for new members that picked that group. Please evaluate the recommendations and explanations **based on whether they are appropriate to users who pick the movie group** (not whether you personally liked the movie).

blood, dark humor, social commentary

Pulp Fiction
Kill Bill: Vol. 1
American History X

Don't know some of these movies? skip to the next group.

Movies	Very Inappropriate 1	2	3	4	Very Appropriate 5	Not Sure
Fargo	○	○	○	●	○	○
Kill Bill: Vol. 2	○	○	○	○	○	○
American Beauty	○	○	○	●	○	○

With minor or no edits, how helpful would these explanations be to display along with the recommended movie for users that pick the movie group?

Explanation	*Not helpful 1*	*2*	*3*	*4*	*Very helpful 5*
A murder story unfolded in a very strange and peculiar way.	○	●	○	○	○
Quirky murder story	○	○	○	○	○

Figure 4: Interface for collecting judgments of recommendations and explanations.

(using an interface similar to figure 2 but part 2 removed). They also write explanations for their recommendations.

2. To prepare for the synthesis step, we find N unique movies from 5 × 5 movies in step 1.
3. Each of the next 5 MovieLens volunteers used interface in figure 2 to synthesize recommendations with the randomly ordered N movies shown as examples (using interface shown in figure 2).
4. We aggregate recommended movies in step 3 to get top 5 most frequent recommendations (if there were ties, we chose randomly), together with explanations on these 5 movies from step 1-3.

For algorithm-assisted pipelines (e.g., **ML_AI**), we replace step 1-2 with the baseline recommendation algorithm, generating the same N number of examples with its human counterpart.

Obtaining Human Quality Judgments

We recruited MovieLens users to provide judgments of the recommendations and explanations produced in the recommendation pipelines. We emailed users who had logged in at least once after November 1, 2014 and who had not been invited to participate in the crowd recommendation experiment. *223* MovieLens users responded to the survey between March 4 and March 29, 2015.

We randomly assigned judges to 1 of of 6 movie groups to evaluate corresponding recommendations and explanations from the 5 pipelines. Judges rated the set of unique movies from the 5 pipelines in random order (shown in figure 4). To avoid the effect of explanations on movie quality judgment, judges can only rate the corresponding explanations after rating a movie.

Figure 4 shows the instructions for judges. Judges rated recommended movies on a five point scale from "Very inappropriate" to "Very appropriate" (mapped to -2 to 2) (with the option to say "Not Sure"), and explanations on a five point scale from "Not helpful" to "Very helpful" (mapped to -2 to 2).

Study: Recommendations

We first ask a simple question: do the different recommendation pipelines actually differ? Do they produce different movies? We then evaluate recommendations produced by the five pipelines using the standard criteria – quality, diversity, popularity, and recency – as well as human judgments. We conclude this section by discussing our results in the context of our research questions. We study the explanations of recommendations given by crowd in next section.

	ML_AI	ML	TK_AI	TK
ML	0.5 (0.96)	-	-	-
TK_AI	1.8 (0.90)	1.3 (0.75)	-	-
TK	0.5 (0.76)	0.5 (0.76)	1 (1)	-
AI	**2.5 (0.69)**	0.8 (0.69)	**2.2 (0.37)**	1 (0.82)

Table 1: Average number of overlapping movies recommended from any pair of pipelines across 6 movie groups. Each pipeline generates 5 movies as final recommendations. Standard deviations are included in parenthesis.

Different pipelines yield different recommendations

Table 1 shows that there is little overlap between the different pipelines. The table shows the average (across the 6 movie groups) number of common movies between each pair of pipelines (each of which generated 5 movies). Recommendations from the two human-only pipelines, **ML** and

TK, have little in common with recommendations from the baseline algorithm: 0.8 common movies for **ML** and 1 common movie for **TK**. The overlap increases for the algorithm-assisted pipelines, with 2.5 common movies **ML_AI** and **AI** and 2.2 common movies between **TK_AI** and **AI**.

Measured Quality: Algorithm slightly better

Our intuition for evaluating quality was: if a pipeline recommends a movie for a specific movie group, how highly would users who like that group rate this movie? We used existing MovieLens data to formalize this intuition: for each movie group, we find the set of MovieLens users who assigned at least one point to the group, and then compute these users' evaluations on recommendations as average of their past ratings on recommended movies.[4]

We compared quality of recommendations from the five pipelines as follows. We used a mixed-effect linear model, in which the pipeline as fixed effect and user as random effect, accounting for the variations of how differently users rate on 5 point scale. Then, we did pairwise comparisons of least squared means of 5 average ratings.

As we expected, recommendations from the baseline algorithm have the highest average rating ($p < 0.001$ using least squared means comparison) as shown in Table 2, because the algorithm is designed to recommend movies of highest average ratings for people who like a movie group. Average ratings of crowd generated recommendations, however, are only slightly worse, less than 0.2 stars on 5-star scale. **TK** is the worst in all crowd pipelines.

	Average Measured Quality (SD)	Average Judged Quality (SD)
AI	**4.19 (0.65)**	0.93 (0.63)
ML	4.12 (0.63)	**1.26 (0.31)**
ML_AI	4.10 (0.66)	1.23 (0.35)
TK	4.00 (0.68)	1.07 (0.68)
TK_AI	4.12 (0.64)	**1.26 (0.31)**

Table 2: Measured and human-judged quality of recommended movies. For a movie group in a pipeline, measured quality is computed as the average of ratings (on a 5 star scale) on recommended movies from MovieLens users who indicated a preference for the movie group. User judgments from the online evaluation range from -2 (very inappropriate) to 2 (very appropriate). Both columns show the average for all pipelines across six movie groups.

Judged Quality: Crowdsourcing pipelines slightly preferred

The human judgments of recommendations gave us another perspective on quality. We analyzed the judgments as follows. We first removed "Not Sure" responses. Then for each judge, we computed their rating for each pipeline by averaging their ratings for the five movies generated by that pipeline. We analyzed the ratings using mixed-effect model similar with that described in the previous section.

Human judges preferred recommendations from all crowd pipelines over the baseline algorithm ($p < 0.01$ using least squared means comparison) as shown in Table 2. (Note that there was larger variance in judgments of recommendations from the algorithm).

There also was an interaction effect between the two experimental factors. Recommendations from algorithm-assisted Turkers (**TK_AI**) were better than those from Turkers only (**TK**). However, there was no differences between the two pipelines involving MovieLens users.

Diversity: A trend for crowdsourcing

Diversity is an attribute of a set of items: how "different" are they from each other. More diverse recommendations help users explore more broadly in a space of items. And prior work showed that recommender system users like a certain amount of diversity (Ziegler et al. 2005; Ekstrand et al. 2014).

We computed the diversity of the 5 movies from each recommendation pipeline using the *tag genome* (Vig, Sen, and Riedl 2012). The tag genome consist of vectors for several thousand movies measuring their relevance to several thousand tags. This lets us compute the similarity of any two movies by computing the similarity of their vectors, for example using cosine similarity. We compute the topic diversity of a set of recommendations as the average pairwise cosine similarities between the tag genome vectors of the recommended movies, a common way to quantify diversity of recommendations (Ziegler et al. 2005). The higher the value, the less diverse the recommendations.

We compared diversity, popularity and recency (described in the following two sections) of recommendations using ANOVA analysis with TukeyHSD test.

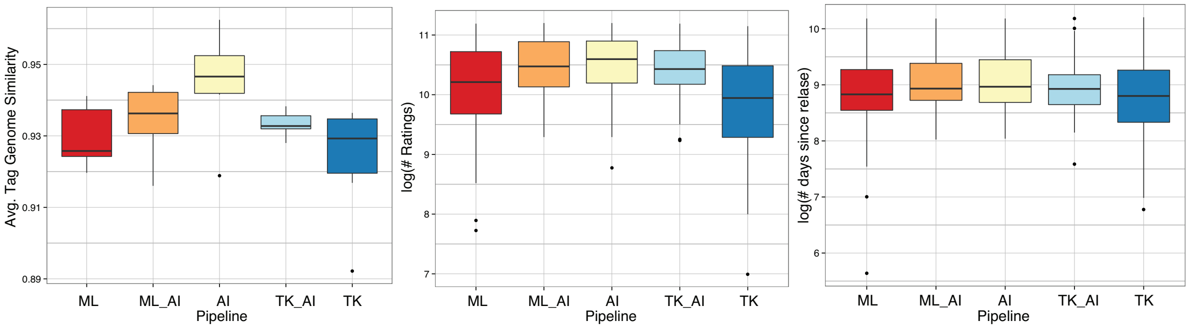
As we had conjectured, crowdsourcing pipelines tend to result in more diverse recommendations (figure 5a). However, the only statistically significant difference was between Turkers in the human-only process (**TK**) and the baseline algorithm ($p < 0.05$). Note that the statistical power for this analysis was reduced because we have fewer data points, just one value per set of recommendations (since diversity is a property of a set of items, not a single item). Thus, we have confidence that that if we had more data, we would find that people generally recommend more diverse movies than the algorithm.

Crowd may give less common recommendations

Prior research(Ekstrand et al. 2014) showed an item's popularity – how frequently it has been rated by users – to be positively correlated with new user satisfaction by earning trust. But popular items are "common knowledge", and thus less likely to be novel to users, and helping users to discover and explore new items is one of the benefits of recommender systems (Vargas and Castells 2011).

We measured movie popularity as the log transform of its number of ratings in MovieLens (this followed a normal distribution after the transformation). Note that this definition

[4] There were a median of 485.5 such users per movie group; min 180, max 780.



(a) Similarity

(b) Popularity

(c) Age

Figure 5: Objective measurements of recommended movies. The middle line represents the median and box boundaries indicate 25th and 75th percentiles. (a) shows average pairwise similarities between tag genome vectors (lower values are more "diverse"), (b) shows number of ratings from all MovieLens users on a natural log scale, and (c) shows number of days since release for movies on natural log scale (lower values are more "recent").

of "popularity" does not mean an item is "liked", just that many users expressed an opinion of it.

Crowdsourcing pipelines tended to result in less popular – thus potentially more novel – movies than the algorithm (see figure 5b). Turkers in the human-only pipeline (**TK**) recommended significantly less popular movies ($p < 0.05$); for MovieLens volunteers, there was a similar trend, but it was not statistically significant.

Recall that workers were more likely to include examples from the algorithm in their recommendation sets than examples from previous workers. That result is reflected in this analysis: for example, Turkers who saw examples from the algorithm (**TK_AI**) were more likely to include them than Turkers who saw examples from previous Turkers (**TK**); this made their final results more similar to the algorithmic baseline results; in particular, they included more popular items.

Recency: Little difference

Prior work showed that different users prefer more or less recent movies in their recommendations (Harper et al. 2015). We measured movie recency as the log transform of the number of days since the movie's release date. The lower the value, the more recent the movie.

Figure 5c suggests that there is a trend for crowdsourcing pipelines to result in more recent movies; however, these differences were not significant.

Discussion

RQ1. We find evidence that the design of the crowdsourcing workflow matters – different pipeline structures lead to different recommendation sets, as measured by overlapping movies. As compared with the algorithmic baseline, the four human pipelines generated more diverse lists that contain less popular movies. We find differences in evaluated quality between the human pipelines and the algorithmic baseline. Looking further into these differences, we find an interesting contradiction: the algorithm-only pipeline generates recommendations with the highest average rating from target users, while the human pipelines are judged to be more appropriate in an online evaluation. We conjecture that the judges are considering user experience factors (e.g., topic match to movie group) that also can impact recommendation quality (McNee, Riedl, and Konstan 2006) — humans recommend movies that are a better fit for the task, but have lower historical ratings.

To summarize the trade-off, using algorithm-generated examples instead of human-generated examples reduced the time required to produce recommendations, but leads to less organic recommendation sets.

RQ3. Perhaps most interesting, we find only small differences in our outcome measures between MovieLens volunteers and Turkers. Since MovieLens members are self-selected as being interested in movies, while Turkers are not, one might conjecture that Turkers are at an enormous disadvantage: movie lovers will recognize more of the movies shown, and will more easily recall related high-quality movies to mind. Possibly we do not find large differences because our recommendation task skews towards mainstream movies (we do not show obscure movies in the movie groups) or because movie watching is such a common pastime. Whether or not this is the case, this is a positive early result for systems wishing to use paid crowdworkers to recommend content.

As mentioned above, we observe an interesting interaction effect where Turkers appear to benefit from algorithmic suggestions more than MovieLens users. It is interesting to speculate why this might be the case. We speculate that this effect for Turkers reveals a relative weakness for recall tasks, along with a strength for synthesis tasks.

Study: Explanations

In this section, we turn our attention to the other output of the CrowdLens process: recommendation explanations. We first analyze the overall quality of explanations, focusing on the differences between paid crowdworkers and MovieLens volunteers (**RQ3**). Then, we analyze language features that are predictive of high quality explanations (**RQ2**).

Explanation quality

The users recruited from MovieLens used our evaluation UI (figure 4) to enter 15,084 ratings (on a scale of -2 to 2) of the quality of 336 crowdsourced explanations. We compare the quality of explanations using a mixed effect linear model, in order to eliminate per-user scoring biases. The model has two independent variables — the type of worker (volunteer or crowd worker) and the source of examples. The model has one dependent variable — average rating of the explanations.

The average rating for explanations from both types of workers is slightly above 0 on a scale of -2 ("Not helpful") to 2 ("Very helpful"). MovieLens users' explanations, on average, received higher ratings than explanations from Turkers (means: 0.14 vs. 0.03, $p < 0.0001$).

There is no significant difference between workers in the human-only and algorithm-assisted pipelines. This is as expected, since showing example movies is not directly related to the task of writing explanations.

Features of Good Explanations

We expect that the quality of explanations provided by crowdworkers will vary substantially — it is natural in crowd work that some contributions will be higher quality than others. In this analysis, we explore the features of explanations that are predictive of high evaluation scores. If we can extract key features of good explanations, we can use this knowledge to provide guidelines for workers as they write explanations.

For this analysis, we labelled explanations with an average rating ≥ 0.25 as "good" (145 explanations are in this class) and those with an average rating ≤ -0.25 as "bad" (102 explanations are in this class). We extracted language features using "Pattern"[5], a popular computational linguistics library. We treated explanations as a bag of words, normalizing words to lowercase and removing common stop words. Based on these features (summarized in table 3), we classify explanations as "good" or "bad" using a logistic regression model.

Qualitatively, we observe substantial variance in the quality of explanations. To inform our discussion of features that are predictive of high and low quality evaluations, let us look at several sample explanations from the study (evaluated "good" or "bad" as described above).

Two examples of "good" explanations:

For "Apollo 13": *Dramatic survival on a damaged space module. Great acting by Tom Hanks, Bill Paxton and Kevin Bacon.*

For "Sleepless in Seattle": *Cute movie of cute kid matchmaking in Seattle and true love upto a the Empire State Building across the country in New York City - so romantic!*

We notice that these (and other good explanations) contain some specific details about the movie (e.g., actors, setting, and plot) and the reasons why the movie is good.

Two examples of "bad" explanations:

For "Apollo 13": *Because is almost exclusively dramatic, good acting and intense.*

For "The Avengers": *It's good vs evil*

We notice that these (and other bad explanations) are too short, overly derivative of the tags shown to describe the movie groups, and not as detailed as the good explanations. Qualitative insights such as these informed the development of features that we subsequently extracted for our regression analysis.

As described above, we use logistic regression to predict "good" explanations, using a broad set of extracted features as input. To evaluate the accuracy of the logistic regression model, we ran a 5-fold cross validation and got a high average F-measure[6] of 0.78. With this high accuracy, we are confident that the extracted features are indicative of the quality of explanations.

Table 3 summarizes the model's features and effects. The model reveals that longer explanations, and explanations containing tags, genres, or other adjectives are more likely to be highly-evaluated.

Discussion

RQ2. Our evaluation revealed that the crowdsourcing pipelines resulted in explanations with neutral to acceptable quality on average, and many that were judged very good. The challenge, therefore, is in selecting and displaying only those explanations with the highest quality. Machine learning methods can predict this measured quality with good accuracy using easily extracted features. We find that the highest-quality explanations tend to be longer, and tend to contain a higher percentage of words that are tags, genres, or other adjectives.

RQ3. In our experiment, MovieLens volunteers provided better explanations than paid Turkers. Likely, this is because the MovieLens volunteers are more familiar with movie recommendations. Also, because they have volunteered to participate in the experiment, they may be more motivated or informed than average users.

Reflection: Volunteers vs. Paid Crowdworkes

A basic decision for crowd-powered projects is where to recruit workers. All projects can recruit from marketplaces like Mechanical Turk. Some can call on volunteers from an existing online community. We explored both options, and our results illuminate key issues.

Time vs. cost. Volunteers may contribute without compensation because of their commitment to the community, but (for small and medium-sized communities), it takes much longer to recruit them in sufficient numbers than to recruit paid crowdworkers. As research-oriented online communities go, MovieLens is among the largest, with a long tradition of members participating in experiments. Yet it took us *1.5 months* to recruit participants for our experiment vs. *1 day* for Turkers. On the other hand, the MovieLens volunteers were free and Turkers are not. Individual projects

[5] http://www.clips.ua.ac.be/pattern

[6] $\frac{2 \cdot precision \cdot recall}{precision + recall}$

Feature	Effect	P-value
log(# words)	1.55	~ 0
% words that appear in tags on movie	2.58	< 0.0005
% adjectives	4.55	< 0.005
% words that appear in genres of movie	5.97	< 0.01
Modality (*-1 to 1 value computed by Pattern to represent uncertain to certain tone*)	0.98	< 0.01
Subjectivity (*0 to 1 value computed by Pattern*)	1.26	0.06
# typos (*given by Google spell checker*)	-1.61	0.07
% words that are directors' names	3.97	0.07
Polarity (*-1 to 1 value computed by Pattern to represent negative to positive attitude*)	0.54	n.s.
% nouns	1.64	n.s.
% verbs	-1.62	n.s.
% words that appear in movie title	-0.02	n.s.
% words that are actor names	-2.88	n.s.
% words that are three tags used to describe the movie group the movie is recommended for	1.75	n.s.
% words that appear in plot summary of the movie	-0.38	n.s.

Table 3: Extracted features of recommendation explanations, along with their effect and statistical significance in a logistic regression model to predict whether an explanation is evaluated to be good or bad by MovieLens users.

must assess the time v.s. cost tradeoff. However, when doing so, they must consider another issue: work quality.

Quality. Volunteers from a community generally have more domain knowledge than paid crowdworkers and some commitment to the community. Thus, one would conjecture that volunteers would produce better work. This is what we found: MovieLens volunteers produced better recommendations and explanations than Turkers. Some Turkers wrote very short explanations or copied-and-pasted excerpts from the Wikipedia page for a movie. However, one other factor must be considered when deciding between volunteers and paid crowdworkers: algorithmic support.

Algorithmic support. Algorithmic techniques for assessing and increasing result quality are necessary for crowdsourcing to be effective: even the basic "output agreement" technique is an example (VonAhn and Dabbish 2008). We saw this, too. When Turkers were provided example recommendations from the MovieLens algorithm, the quality of their recommendations became comparable to MovieLens volunteers. And while we did not run this experiment, the model of features of high quality explanations could be applied directly to offer guidelines to people writing explanations and automatically assess and critique a written explanation; for example "Your explanation has a 60% chance to be evaluated positively; to improve it, consider writing a longer explanation that describes more attributes of the movie that would make someone interested in watching it".

Thus, to summarize: if you are planning a crowdsourcing effort and have access to site volunteers as well as paid crowdworkers, you should consider the time it would take to recruit sufficient volunteers, the knowledge required to produce acceptable results, and the algorithmic support tools available to assist workers.

Limitations and Future Work

This work is an early exploration of crowdsourcing in recommendation. It did not address all issues in sufficient depth, and it suggests a number of promising opportunities for future research.

First, several deeper evaluations of crowd-powered recommendation are possible. We asked human judges to evaluate individual recommended movies instead of comparing recommendations sets from different pipelines. Comparing (typically overlapping) sets of items on multiple dimensions (quality, diversity, etc.) is a difficult cognitive task. However, recent research successfully experimented with techniques to ease this task, such as highlighting items unique to each set (Harper et al. 2015). Thus, a logical next step would be a field study in which human judges evaluate sets of recommendations from different crowdsourcing pipelines and a baseline algorithm. Further, while we asked judges only to evaluate the effectiveness of explanations, other factors such as transparency, trust, and persuasiveness and other factors have been studied (Tintarev and Masthoff 2012). Thus, a study that gathered multidimensional assessments of recommendation explanations – both automatically generated and produced by people – would be interesting.

Second, we saw that humans can benefit from algorithmically generated examples; can algorithms also benefit from human-produced recommendations? We saw that human-produced recommendation sets tended to be more diverse and feature more potentially novel items. These recommendation sets could be used as input to a learning algorithm, for example, to produce an ensemble recommender that combined output from people and the baseline recommendation algorithm.

Third, crowdworker performance can be influenced by many factors such as compensation amount, previous experience, demographics, etc. We can study in more depth about how these factors affect the outcome of crowd recommendations.

Summary

We explored a novel research problem: how to incorporate human wisdom into the recommendation process. We experimented with two different workflows and two types of participants: volunteers from the MovieLens web site and paid crowdworkers from Amazon Mechanical Turk. We found that crowdsourced workflows resulted in recommendations that were higher evaluated by users than a state-of-the-art recommendation algorithm and that tended to be more diverse and include less common (thus potentially more novel) items. Volunteers produced better recommendations than paid crowdworkers; however, when the crowdworkers were provided with algorithmically generated examples, this gap disappeared (albeit at the cost of reduced diversity). The crowdsourcing process resulted in many recommendation explanations that were judged of high quality; again, volunteers performed better than paid crowdworkers. We identified features of good explanations and a model that can predict high-quality explanations with good accuracy. Finally, we reflected in depth on the tradeoffs in recruiting volunteers vs. paid crowdworkers and identified a number of rich topics for future research.

Acknowledgments

We thank anonymous reviewers for their feedback and NSF for funding our research with awards IIS-0968483, IIS-0964695, IIS-1218826, IIS-1210863 and IIS-1017697.

References

Bernstein, M. S.; Little, G.; Miller, R. C.; Hartmann, B.; Ackerman, M. S.; Karger, D. R.; Crowell, D.; and Panovich, K. 2010. Soylent. In *UIST*.

Chang, S.; Harper, F. M.; and Terveen, L. 2015. Using Groups of Items to Bootstrap New Users in Recommender Systems. In *CSCW*.

Ekstrand, M. D.; Harper, F. M.; Willemsen, M. C.; and Konstan, J. A. 2014. User perception of differences in recommender algorithms. In *Recsys*.

Felfernig, A.; Haas, S.; Ninaus, G.; Schwarz, M.; Ulz, T.; Stettinger, M.; Isak, K.; Jeran, M.; and Reiterer, S. 2014. RecTurk: Constraint-based Recommendation based on Human Computation. In *Recsys 2014 - CrowdRec Workshop*.

Harper, F. M.; Xu, F.; Kaur, H.; Condiff, K.; Chang, S.; and Terveen, L. 2015. Putting Users in Control of their Recommendations. In *RecSys*.

Herlocker, J. L.; Konstan, J. A.; and Riedl, J. 2000. Explaining collaborative filtering recommendations. In *CSCW*.

Kittur, A.; Smus, B.; Khamkar, S.; and Kraut, R. E. 2011. CrowdForge. In *UIST*.

Krishnan, V.; Narayanashetty, P. K.; Nathan, M.; Davies, R. T.; and Konstan, J. A. 2008. Who predicts better? In *RecSys*.

Kulkarni, C.; Dow, S.; and Klemmer, S. 2014. Early and repeated exposure to examples improves creative work. 49–62.

Lasecki, W. S.; Wesley, R.; Nichols, J.; Kulkarni, A.; Allen, J. F.; and Bigham, J. P. 2013. Chorus: A Crowd-powered Conversational Assistant. In *UIST*.

Little, G.; Chilton, L. B.; Goldman, M.; and Miller, R. C. 2009. TurKit. In *HCOMP*.

McNee, S. M.; Riedl, J.; and Konstan, J. A. 2006. Being accurate is not enough. In *CHI EA*.

Nebeling, M.; Speicher, M.; and Norrie, M. C. 2013. CrowdAdapt: Enabling Crowdsourced Web Page Adaptation for Individual Viewing Conditions and Preferences. In *EICS*.

Organisciak, P.; Teevan, J.; Dumais, S.; Miller, R. C.; and Kalai, A. T. 2014. A Crowd of Your Own: Crowdsourcing for On-Demand Personalization. In *HCOMP*.

Retelny, D.; Robaszkiewicz, S.; To, A.; Lasecki, W. S.; Patel, J.; Rahmati, N.; Doshi, T.; Valentine, M.; and Bernstein, M. S. 2014. Expert Crowdsourcing with Flash Teams. In *UIST*.

Siangliulue, P.; Arnold, K. C.; Gajos, K. Z.; and Dow, S. P. 2015. Toward collaborative ideation at scale: Leveraging ideas from others to generate more creative and diverse ideas. In *CSCW*.

Smith, S.; Ward, T.; and Schumacher, J. 1993. Constraining effects of examples in a creative generation task. *Memory and Cognition* 21(6):837–845.

Tintarev, N., and Masthoff, J. 2012. Evaluating the effectiveness of explanations for recommender systems. *User Modeling and User-Adapted Interaction* 22:399–439.

Vargas, S., and Castells, P. 2011. Rank and relevance in novelty and diversity metrics for recommender systems. Recsys.

Vig, J.; Sen, S.; and Riedl, J. 2012. The Tag Genome. *ACM Transactions on Interactive Intelligent Systems* 2(3):1–44.

VonAhn, L., and Dabbish, L. 2008. Designing games with a purpose. *Communications of the ACM* 51(8):58–67.

Xu, A.; Rao, H.; Dow, S. P.; and Bailey, B. P. 2015. A Classroom Study of Using Crowd Feedback in the Iterative Design Process. In *CSCW*.

Xu, A.; Huang, S.-W.; and Bailey, B. 2014. Voyant: Generating Structured Feedback on Visual Designs Using a Crowd of Non-experts. In *CSCW*.

Zhang, H.; Law, E.; Miller, R.; Gajos, K.; Parkes, D.; and Horvitz, E. 2012. Human computation tasks with global constraints. In *CHI*.

Ziegler, C.-N.; McNee, S. M.; Konstan, J. A.; and Lausen, G. 2005. Improving recommendation lists through topic diversification. In *WWW*.