



CrowdStart: Warming up cold-start items using crowdsourcing

Dong-Gyun Hong^{a,1}, Yeon-Chang Lee^{a,1}, Jongwuk Lee^{b,*}, Sang-Wook Kim^{a,*}

^a Hanyang University, Seoul, Republic of Korea

^b Sungkyunkwan University, Suwon, Republic of Korea



ARTICLE INFO

Article history:

Received 18 July 2018

Revised 6 June 2019

Accepted 12 July 2019

Available online 15 July 2019

Keywords:

Collaborative filtering

New item recommendation

Crowdsourcing

ABSTRACT

The cold-start problem is one of the critical challenges in personalized recommender systems. A lot of existing work has been studied to exploit a user-item rating matrix as well as additional information for users/items, e.g., user profiles, item contents, and social relationships among users. However, because existing work is primarily biased to the auxiliary information for users/items, it is difficult to identify various and reliable item neighbors that are relevant to cold-start items. To alleviate this limitation, we propose a new *crowd-enabled* framework, called CrowdStart, which is an integrated *human-machine* approach for new item recommendation. The main contributions of the CrowdStart framework are two-fold: (1) To find various and reliable item neighbors for new items, we design two-step crowdsourcing tasks that harness not only machine-only algorithms but also the knowledge of crowd workers (including a few experts and a large number of non-expert workers in a crowdsourcing platform). (2) We develop a novel hybrid model to exploit the user-item rating matrix, the content information about items, and the crowd-based item neighbors from human knowledge into new item recommendation. To evaluate the effectiveness of the CrowdStart framework, we conduct extensive experiments including both a user study and simulation tests. Through the empirical study, we found that the CrowdStart framework provides *relevant*, *diverse*, *reliable*, and *explainable* crowd-based neighbors for new items and the crowd-based neighbors are meaningful for improving the accuracy of new item recommendation. The datasets and detailed experimental results are available at <https://goo.gl/1iXTUE>.

© 2019 Elsevier Ltd. All rights reserved.

1. Introduction

Collaborative filtering (CF) (Adomavicius & Tuzhilin, 2005; Ricci, Rokach, & Shapira, 2015) is known as one of the successful techniques for personalized recommender systems. The key idea of CF is that a user u is likely to prefer an item i if another user v with similar preference patterns to u also prefers the item i . Commonly, the similarity between users is measured with the items co-rated by them. Because CF leverages a *user-item rating matrix*, it can be applied to any application domain. However, the main drawback of CF is that, as the degree of *data sparsity* increases, the accuracy of top- N recommendation decreases. In particular, when an item i has few or no ratings, it is much difficult to recommend it to any user, which is well-known as the *cold-start* problem (Adomavicius & Tuzhilin, 2005; Lika, Kolomvatsos, & Hadjiefthymiades, 2014).

To address this problem, a lot of existing work (Guo, Zhang, & Yorke-Smith, 2015; He, Lin, Wang, & J., 2016; He & McAuley, 2016a; 2016b; Jamali & Ester, 2009; 2010; Kim, Park, Oh, Lee, & Yu, 2016; Li, Kawale, & Fu, 2015; Ma, Lyu, & King, 2009; Ma, Yang, Lyu, & King, 2008; Ma, Zhou, Liu, Lyu, & King, 2011; Park & Chu, 2009; Tang, Aggarwal, & Liu, 2016; Wang & Blei, 2011; Wang, Wang, & Yeung, 2015; Yao, He, Wang, Zhang, & Cao, 2015; Ying, Chen, Xiong, & Wu, 2016) has been studied to exploit a user-item rating matrix as well as additional information about users/items. The auxiliary information is categorized into three types: (1) *textual contents* of items such as genres and movie descriptions (Kim et al., 2016; Li et al., 2015; Park & Chu, 2009; Wang & Blei, 2011; Wang et al., 2015; Yao et al., 2015; Ying et al., 2016), (2) *visual information* of items such as product images (He et al., 2016; He & McAuley, 2016a; 2016b), and (3) *social relationships* among users (Guo et al., 2015; Jamali & Ester, 2009; 2010; Ma et al., 2009; Ma et al., 2008; Ma et al., 2011; Tang et al., 2016). Existing work has been studied to capture the hidden features of cold-start items/users. However, it is non-trivial to identify various and reliable item neighbors that are relevant to cold-start items.

* Corresponding authors.

E-mail addresses: dg4271@agape.hanyang.ac.kr (D.-G. Hong), lyc0324@hanyang.ac.kr (Y.-C. Lee), jongwuklee@skku.edu (J. Lee), wook@hanyang.ac.kr (S.-W. Kim).

¹ This paper has two first authors who contributed equally to this work.



Fig. 1. Neighbors relevant to a new item “Wonder Woman” obtained by CrowdStart, LDA, CNN, CTR, and ConvMF. (Crowd-based neighbors include “Catwoman,” “Suicide Squad,” “Thor,” and “The Avengers”; LDA-based neighbors include “Kim,” “Kull the Conqueror,” “Macbeth,” and “The Wolf Man”; CNN-based neighbors include “Drugstore Cowboy,” “Whores’ Glory,” “The State of Things,” and “The Shooter”; CTR-based neighbors include “Hell in the Pacific,” “Instinct,” “Dark Tide,” and “The Messenger”; ConvMF-based neighbors include “Little Odessa,” “A Funny Thing Happened on the Way to the Forum,” “Arthur and the Invisibles,” and “The Gun in Betty Lou’s Handbag.”)

Fig. 1 depicts the examples of item neighbors for the movie “Wonder Woman”² obtained from existing work such as *latent Dirichlet allocation* (LDA), *convolutional neural network* (CNN), *collaborative topic regression* (CTR) (Wang & Blei, 2011) and *convolutional matrix factorization* (ConvMF) (Kim et al., 2016). We observe that existing algorithms fail to capture the overall storyline of movies by only considering textual information and rating information.

Essentially, machine-only algorithms are difficult to capture the comprehensive understanding of the hidden features of items. To address this limitation, the expert systems (Jackson, 1986) addressed the complex problem by utilizing human knowledge from a few domain experts. For instance, existing work performs a decision making process such as prediction (Wang, 2002) and monitoring (Zhou, W., & P., 2009) by fusing the human knowledge in inference engines.

Motivated by this idea, we make use of human knowledge to solve the cold-start problem. Unlike existing expert systems that utilize a few domain experts, we adopt *crowdsourcing* which is a new computing paradigm to integrate human and machine computations. It enables us to leverage various crowd workers including a few experts and a large number of non-expert workers via a web service such as *Amazon Mechanical Turk* (MTurk) and *Figure Eight*. In existing work (Chang, Amershi, & Kamar, 2017; Ghosh, Sharma, Benevenuto, Ganguly, & Gummadi, 2012; Ipeirotis & Gabrilovich, 2014; Mohit, Singh, & Gurpreet, 2016; Said et al., 2014; Yu, Xu, Yang, & Guo, 2016), crowdsourcing has been successfully performed for complex and non-trivial tasks that require intuitive human knowledge.

In this paper, we propose a novel crowd-enabled hybrid framework, called *CrowdStart*, to alleviate the cold-start problem by incorporating the rating-based user similarities, the machine-based item neighbors, and the crowd-based item neighbors into new

item recommendation. Our intuition of using crowdsourcing is as follows:

- **Filling various similarity holes:** Crowd workers can identify various item neighbors for the new item based on their criteria. By collecting various perspectives for the new item from crowd workers, we can obtain *various* crowd-based neighbors for the new item.
- **Overcoming limited content analysis:** Crowd workers are capable of associating various information about items such as text, images, and videos. They can provide reasonable *explanations* for the similarity between neighbors and the new item. Therefore, we can obtain *relevant*, *reliable*, and *explainable* neighbors for the new item.

The *CrowdStart* framework consists of an integrated *human-machine* approach. The role of the human (*i.e.*, crowd workers) is to build a large and diverse knowledge base at a low cost and to filter noisy information.³ Specifically, we utilize the *wisdom of crowds* to obtain item neighbors that are somehow “similar” to a new item. Next, the role of the machine is to build an inference engine using the knowledge obtained from the crowd workers and to recommend the new item to users based on the inference engine. Here, we employ a neighbor-based CF model as an inference engine for new item recommendation.

To utilize the wisdom of crowds, we design two crowdsourcing tasks, *i.e.*, *generation and voting tasks*, executed by crowd workers. First, the generation task is used to collect as many candidates of crowd-based neighbors for a new item as possible. We ask crowd workers to generate items relevant to the new item. Based on their feedback, we can obtain *diverse* item candidates. Then, the items

³ Because the recommendation problem (*e.g.*, movie) does not require significant expertise, we believe that crowdsourcing is suitable for solving the recommendation problem.

² Note that this movie has been not released yet in conducting our experiments.

are used together with other item candidates obtained from existing machine-only algorithms that utilize additional information.

Second, the voting task is used to select *reliable* neighbors by filtering irrelevant item candidates. It is necessary to remove noisy information from the item neighbors obtained from the crowd workers and the machine-only algorithms. Toward this goal, we request crowd workers to click on relevant items among the candidates for a new item and determine reliable neighbors for the new item based on the number of clicks per item. In this process, we can also ask workers to fill in reasonable *explanations* for their votes. Finally, we can collect the crowd-based neighbors for the new item, as in Fig. 1(b). It is found that crowd-based neighbors include *diverse*, *relevant*, and *reliable* movies compared to other neighbors by considering general features, e.g., the heroine, characters in the original comic, and storyline of "Wonder Woman."

After collecting crowd-based neighbors for the new item, we lastly retrieve a set of users who are interested in the crowd-based neighbors. We generate a *pseudo user* who has rated both crowd-based neighbors and the new item. Then, we calculate the similarity between users and the pseudo user and identify the users who have similar rating patterns to the pseudo user. The users can be highly likely to be interested in the new item. Finally, we recommend the new item to the users. Note that the goal of this paper is not to predict how much users would like a new item, but to find the users who are likely to be interested in the new item. Because the crowd-based neighbors are collected by the implicit feedback (i.e., rather than explicit ratings) of crowd workers, it is possible to infer users' interest (not relative preferences) for the new item. Likewise, because the CrowdStart framework does not predict the explicit ratings for the new items, we cannot distinguish the relative preferences between new items and existing items. In other words, the CrowdStart framework aims at informing users who are likely to be interested in the new item, whenever creating a new item, in addition to providing the recommendation for existing items.

To evaluate the effectiveness of the CrowdStart framework, we conduct both an extensive user study and simulation tests. Specifically, we examine two core steps of the CrowdStart framework: crowd-based neighbors and new item recommendation. To validate crowd-based neighbors, we compare them against the following two types of neighbors obtained from existing algorithms: (1) content-based neighbors obtained from existing content-based algorithms such as LDA and CNN; (2) rating-based neighbors with additional information obtained from hybrid algorithms such as CTR and ConvMF. We also evaluate new item recommendation using simulation tests. We select the top- k users who are likely to be interested in the new item and check if they are interested in it.

In summary, the contributions of this paper are as follows:

- We address how to integrate crowdsourcing into a hybrid recommendation mechanism for new item recommendation. To the best of our knowledge, this paper is the first attempt that utilizes crowdsourcing to alleviate the cold-start item problem.
- We propose the CrowdStart framework which can (1) identify relevant, diverse, reliable, and explainable crowd-based neighbors, and (2) provide useful new item recommendation using the rating-based user similarities, the machine-based item neighbors, and the crowd-based item neighbors.
- We validate the effectiveness of the CrowdStart framework via a user study and simulation tests. We evaluate the core parts of selecting crowd-based neighbors and providing new item recommendation in the CrowdStart framework.

The rest of this paper is organized as follows. In Section 2, we overview existing work for the cold-start problem and crowd-

enabled recommendations. In Section 3, we present the overall procedure of the CrowdStart framework and its implementation details. In Section 4, we empirically evaluate the effectiveness of the CrowdStart framework through user study and simulation tests. In Section 5, we finally conclude this paper.

2. Related work

In this section, we present existing work to overcome the cold-start problem and review several studies using crowdsourcing for recommender systems.

2.1. The cold-start problem in recommendation

Cold-start item recommendation. Park and Chu (2009) proposed a *regression model* using some content information of items such as genre, cast, manufacturer, and release year. Similarly, Wang and Blei (2011) and Yao et al. (2015) exploited *textual* information of items such as movie descriptions and reviews. They adopted a topic model using *latent Dirichlet allocation* (LDA) to analyze textual information of items and combined it with matrix factorization.

Recently, deep neural networks (Kim et al., 2016; Li et al., 2015; Wang et al., 2015; Wu et al., 2018; Wu, DuBois, Zheng, & Ester, 2016; Ying et al., 2016; Zheng, Noroozi, & Yu, 2017) have been used to analyze textual information. Several studies (Li et al., 2015; Wang et al., 2015; Wei, He, Chen, Zhou, & Tang, 2017; Wu et al., 2016; Ying et al., 2016) leveraged *denoising auto-encoders* (DAE) to capture the semantics of textual information. Likewise, Kim et al. (2016) and Zheng et al. (2017) utilized a *convolutional neural network* (CNN) to analyze the underlying semantics of textual information. Furthermore, Wu et al. (2018) proposed dual-regularized matrix factorization (DRMF) framework exploiting CNN and *recurrent neural network* (RNN).

Besides, because the CNN model is effective at capturing latent *visual* features of images, it has been actively used for understanding the visual information of product images. Images are embedded in latent spaces to identify items with high visual similarities to optimize relative rankings of items. Specifically, VBPR (He & McAuley, 2016b) and DeepStyle (Liu, Wu, & Wang, 2017) combined the CNN model with *Bayesian personalized ranking* (BPR). TVBPR (He & McAuley, 2016a) extended VBPR (He & McAuley, 2016b) by considering *temporal* features of items. Sherlock (He et al., 2016) proposed a *sparse hierarchical embedding* method to discover both globally-relevant dimensions of items and subtle local dimensions for visual information. VPOI (Wang et al., 2017) also used CNN to extract hidden features from images and addressed the cold-start user problem by exploiting the images posted by users.

Cold-start user recommendations. To address the cold-start user problem, previous studies utilized *social* information such as trust/friend relationships between users. Jamali and Ester (2009) proposed a *random walk* model that combines the trust relationships between users with collaborative filtering. In addition, several studies (Guo et al., 2015; Guo, Zhang, Zhu, & Wang, 2017; Jamali & Ester, 2010; Lin, Zhang, Zhang, Liu, & Ma, 2017; Ma et al., 2009; Ma et al., 2008; Ma et al., 2011; Moradi & Ahmadian, 2015; Rafailidis & Crestani, 2017; Shi, Wu, Zheng, Shi, & Li, 2017; Tang et al., 2016) exploited the trust relationships between users as auxiliary information to build accurate *user features*. In other words, they optimized user and item matrices by reflecting both the trust relationships between users and user-item ratings.

In summary, the previous studies for cold-start items/users attempted to further analyze the hidden features of items/users using additional information. Specifically, early work employed simple machine learning algorithms such as LDA, which are, however,

is limited to analyze complex and non-linear relationships. In recent years, deep-learning-based algorithms such as CNN and RNN have been adopted for recommender systems. However, they are still limited for considering reliable and diverse relationships to find latent similarities between items and users by using additional information. In clear contrast, we explore the idea of *crowdsourcing* for the cold-start item problem. That is, our key contributions are to exploit various and reliable knowledge from crowd workers (not machine-only algorithm) to capture underlying similarities among items.

2.2. Recommendation with crowdsourcing

Recently, expert systems adopt a large number of crowd workers in a crowdsourcing platform to perform some non-trivial tasks such as search (Ghosh et al., 2012), knowledge management (Ipeirotis & Gabrilovich, 2014), and labeling (Chang et al., 2017). Specifically, Ghosh et al. (2012) proposed a new system, called Cognos, for discovering topic experts in microblogs such as Twitter. Cognos yielded the results comparable to those given by the official Twitter experts. Ipeirotis and Gabrilovich (2014) developed a gamified crowdsourcing system, called Quizz, that simultaneously evaluates the knowledge of users and acquires new knowledge from them. Recently, Chang et al. (2017) designed a collaborative approach, called Revlot, that brings ideas from expert annotation workflows to crowd-based labeling. They demonstrated that the proposed model produces high-quality labels without requiring label guidelines. Although some particular tasks are more suitable for a few experts, we found that many tasks are sufficient by using crowd workers. Based on this idea, we study how to leverage crowdsourcing for new item recommendations.

Some recent studies have leveraged crowdsourcing to address some issues in recommender systems: (1) alleviation of data sparsity, (2) recommendation for new users, and (3) explanations for recommendation results.

Alleviation of data sparsity. Because the central challenge of recommender systems arises from *data sparsity*, it is critical to collect more information for items/users. Lee et al. (2013) proposed a crowdsourcing-based collaborative filtering method that solicited new user ratings from crowd workers and augmented a user-item rating matrix. Felfernig et al. (2014, 2015) used crowdsourcing to obtain diverse knowledge for the *constraint-based* recommendation. Said et al. (2014) designed a recommender system, namely CrowdRec, that actively interacts with the crowd by considering the concept of ‘incentivization.’ Yu et al. (2016) proposed constructing the user profile and location model based on crowd-sourced check-in records for personalized travel package recommendation. Mohit et al. (2016) selected the crowd who is close to the location of the particular user using k-means clustering and aggregated their preferred items to provide the recommendation. Finally, Smyth, Rafter, and Banks (2016) explored a *game-with-a-purpose* (GWAP) to collect recommendation results and user preferences as a side effect of casual gameplay. They showed that augmented data obtained from crowd workers are useful for improving the accuracy of recommendations.

Recommendations for new users. Because it is inherently difficult to capture latent preferences of *new users*, recent studies (Chang, Harper, He, & Terveen, 2016; Organisciak, Teevan, Dumais, Miller, & Kalai, 2014, 2015) have leveraged crowdsourcing to infer the preferences of new users. Organisciak et al. (2014, 2015) proposed two protocols for crowdsourcing tasks: *taste-matching* for identifying crowd workers who have similar preferences to a requester, and *taste-grokking* for asking crowd workers to predict the requester’s preference. Chang, Harper, He et al. (2016) proposed various workflows that integrate crowdsourcing into the new user recommendation loop. They showed

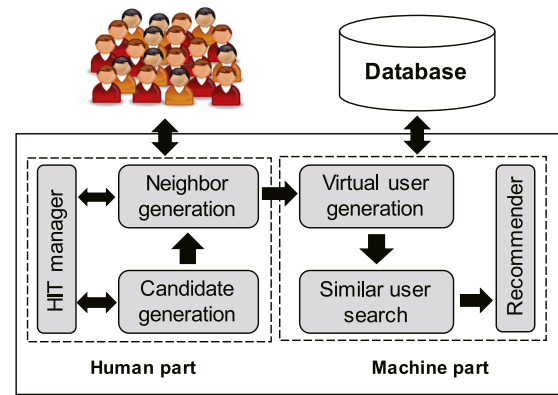


Fig. 2. Overall process of the CrowdStart framework.

that crowdsourcing is effective at indirectly discovering the latent preferences of new users.

Explanations for recommendation results. Providing *reasonable explanations* for recommendation results is non-trivial. Chang, Harper, and Terveen (2016) exploited crowdsourcing to obtain natural language explanations for recommendation results. Because crowd workers can capture the latent similarities between movies, it is shown that the explanations obtained from crowd workers can be more persuasive and trustworthy than explanations generated by existing algorithms.

3. Proposed framework

In this section, we propose a novel crowd-enabled framework, called CrowdStart. We explain the key process of the CrowdStart framework: (1) how to leverage crowd workers for new item recommendation, (2) how to implement new item recommendations in the crowdsourcing platform, and (3) how to design a user interface for crowd workers.

3.1. Overall description

The goal of the CrowdStart framework is to provide personalized recommendation systems for a new item. Toward this goal, the CrowdStart framework is designed by human-machine collaboration. Fig. 2 describes the overall process of the CrowdStart framework. The human part in the CrowdStart framework is involved in two crowdsourcing tasks (i.e., generating and voting on items relevant to the new item), and these are executed by crowd workers and machine-only algorithms. After the items relevant to the new item are selected, the machine part is used to identify a set of users who are interested in the new item. The CrowdStart framework consists of three main steps: (1) *generating some diverse candidate neighbors for the new item*, (2) *voting on the most reliable items among them to determine crowd-based neighbors*, and (3) *incorporating rating-based user similarities and a set of items selected by crowd workers into new item recommendations*.

First, we explain the two crowdsourcing steps performed in the human part. Specifically, they identify a set of candidate items that are relevant to a new item. The *relevance* is usually measured by the similarity between items. That is, given two relevant items i and j , the user likes item i if she is also likely to be interested in item j (and vice versa). It is because they are similar to each other in terms of the user’s preference. Given item i , its *neighbors* are defined as a set of items that are relevant to i . Also, *crowd-based neighbors* of the new item are defined as a set of neighbors that are determined by various preferences of crowd workers.

Step 1: Generating candidate items. The new target item is first taken as input. Then, the CrowdStart framework asks crowd

workers to generate items thought to be relevant to the new item with the reasons why they are similar. In this process, content descriptions including textual and visual information on the new item are provided to crowd workers. Because relevance is to be defined by the different intuitions of crowd workers, the latent similarities between items can be captured in different ways, making neighbors obtained from workers diverse. Also, the workers provide explanations about their choices. In this process, because the task requires the workers to answer a broad question, the crowd-sourced results may be unpredictable. To resolve this problem, we append the neighbors generated by existing machine-only algorithms that use additional information. In other words, we can collect *diverse* and *explainable* candidate neighbors that are relevant to the new item by complementing both crowd workers and machines.

Step 2: Voting on the most reliable items. The candidate neighbors may still contain some items irrelevant to the new item. We thus perform a *filtering process* using crowd workers. A set of candidate items collected by crowd workers and machine-only algorithms is taken as input. Then, the CrowdStart framework asks crowd workers to choose the most relevant items among candidate neighbors by clicking on them, with explanations for their votes. To filter unnecessary items out, we sort candidate items by the click counts. Then, we consider the remaining items as noisy items, except for the top- N items that received the most votes. Finally, by removing noisy items, we determine *reliable* and *explainable* neighbors for the new item. The items thus determined by crowd workers as neighbors to a new item are called *crowd-based neighbors* for the new item.

Step 3: Incorporating crowd-based neighbors. After the crowd-based neighbors are obtained from the human part, the CrowdStart framework attempts to recommend the new item to a set of top- k users using the neighbors. To do this, we first generate a *virtual user* v who is interested in both crowd-based neighbors and the new item. Given the virtual user, we identify a set of users in a database who have similar tastes to her/him. To do this, we calculate the similarity between a user u and a virtual user v using binary cosine similarity.⁴ Specifically, we implicitly represent the item preferences of the two users as binary vectors (*i.e.*, one-class setting) because the crowd-based neighbors are generated from the implicit feedback of crowd workers. Based on the calculated similarity, we identify the top- k similar users and then designate these users having similar tastes to the virtual user. Because the top- k users are highly likely to be interested in the new item, the CrowdStart framework recommends the new item to them.

In the CrowdStart framework, the goal of the new item recommendation process is to inform users who are likely to be interested in a new item whenever it is created. That is, we assume that new items are recommended *independently* of the existing item recommendations. In this process, we can additionally ask crowd workers to check whether a target user likes the new item, as done in taste-matching and taste-grokking (Organisciak, Teevan, Dumais, Miller, & Kalai, 2015). However, because it requires us to perform a validation task for each user, it can incur too much cost. In this sense, crowdsourcing is only used for identifying crowd-based neighbors for the new item in our proposal.

3.2. Crowdsourcing tasks

To implement the CrowdStart framework, we employ the popular crowdsourcing platform, *Amazon Mechanical Turk (MTurk)*. In general, MTurk is composed of three main parts:

- **Requester:** The requester is an individual or a group that publishes a crowdsourcing task. S/he is responsible for paying monetary rewards to crowd workers.
- **Worker:** The crowd worker is an internet user who performs a specific task. After performing a task, s/he can receive rewards from the requester.
- **Task:** The task is a question asked to crowd workers, called a *human intelligence task (HIT)*. Various tasks can be formulated depending on the purpose of the requester. The CrowdStart framework performs two tasks: a *generation task* and a *voting task*.

To acquire diverse candidate neighbors for a new item, we first create a HIT for the generation task. Fig. 3 shows the user interface of the generation task: (A) showing a guideline with examples; (B) providing the details of a new item; (C) inputting relevant items. Its format is based on a *subjective question*. Given a target item, crowd workers fill in relevant items along with some reasons. In this process, if the answers of workers are insufficient, we regard the workers as spam workers and then manually filter their responses out.⁵ Then, we store the remaining answers of workers as candidate neighbors for the target item. We note that, because this task is based on a subjective question, record identification and noise problems (*e.g.*, typos) may occur. To alleviate the problem, we manually perform pre-processing of the answers obtained from workers. After completing the generation task, we also employ some neighbors generated by the existing machine-only algorithms.

Then, we create a HIT for the voting task based on the results of the generation task. Fig. 4 shows the user interface of the voting task: (A) showing a guideline with examples; (B) providing the details of a new item and candidate neighbors; (C) clicking relevant items. The format of the HIT follows a *multi-ary question*. That is, we ask crowd workers (different from workers in the voting task) to click on relevant items out of candidate neighbors and to input the reason why the items are related to the target item. In this process, we count on each item and manually filter answers from spam workers in the same way as the generation task. After that, we determine the top- N items with the highest number of votes as crowd-based neighbors. Finally, we recommend the target item to a set of users who have shown a preference for its crowd-based neighbors.

Note that the accuracy of our framework highly depends on the quality of the results obtained from the crowdsourcing tasks. In existing work for the crowdsourcing problem (Garcia-Molina, Joglekar, Marcus, Parameswaran, & Verroios, 2016; Guo, Wang, Song, & Hong, 2014; Li, Wang, Zheng, & Franklin, 2016), worker management and task assignment are known as essential issues to control the quality of results. In other words, all the workers do not have the same quality, and all the tasks do not have the same difficulty.

To address these issues, we design various features in our tasks. For worker management, the following features are considered: (1) in all tasks, we use the qualification requirement of a *HIT approval rate* and *number of HIT's approved* on MTurk; (2) in the generation and voting tasks, we provide a *specific keyword* only to the workers who answered all the questions of the task, and ask to enter the keyword as the last question. Then, we manually regard workers who fail to enter the keyword (*i.e.*, workers who have not answered all the questions) as *untrustworthy workers*; (3) in the voting task, we inject several *fake movies* and manually regards workers as *spam workers* if they select the fake movies. For task assignment, we set *different costs* according to tasks. Because the

⁴ It is also possible to use other similarity measures.

⁵ We will present details on how to distinguish spam workers in the descriptions for worker management.

Write the movies that you think is relevant. (Click to collapse)

Given a movie, please find other relevant movie titles by reading the guidelines and movie information given below.

Guidelines:

- You must fill out at least two movie titles for a given movie.
- Write the reason why those movies are relevant.
- The example of the movie relevance: the similarity of synopsis, the similarity of genres.

*** If you give incorrect or inappropriate answers, we may not compensate for rewards.**

Movie informations:

- Movie trailer: [Movie trailer url](#)

- Title: Wonder Woman

- Genres: Action,Adventure,Fantasy

- Director: Patty Jenkins

- Stars: Gal Gadot,David Thewlis,Robin Wright

- Plot summary:

- Before she was Wonder Woman, she was Diana, princess of the Amazons, trained to be an unconquerable warrior. Raised on a sheltered island paradise, when an American pilot crashes on their shores and tells of a massive conflict raging in the outside world, Diana leaves her home, convinced she can stop the threat. Fighting alongside man in a war to end all wars, Diana will discover her full powers and her true destiny.
- Before she was Wonder Woman she was Diana, princess of the Amazons, trained warrior. When a pilot crashes and tells of conflict in the outside world, she leaves home to fight a war to end all wars, discovering her full powers and true destiny.

More information: [IMDB information url](#)

<p>Movie title 1:</p> <input style="width: 90%;" type="text"/>	<p>Movie title 2:</p> <input style="width: 90%;" type="text"/>	<p>Movie title 3:</p> <input style="width: 90%;" type="text"/>
<p>Explanation 1:</p> <div style="border: 1px solid #ccc; height: 80px; width: 100%;"></div>	<p>Explanation 2:</p> <div style="border: 1px solid #ccc; height: 80px; width: 100%;"></div>	<p>Explanation 3:</p> <div style="border: 1px solid #ccc; height: 80px; width: 100%;"></div>

Fig. 3. The user interface of the generation task with the following parts-(A) showing a guideline with examples, (B) providing the details of a new movie, and (C) inputting related movies.

generating task (*i.e.*, step 1) is more time-consuming and complicated than the voting task (*i.e.*, step 2), we provide more cost to the workers performing the former.

3.3. Discussion

To summarize, the main contributions of the proposed framework are two-fold. First, to find the various and reliable item neighbors for new items, we perform two-step crowdsourcing tasks that utilize knowledge of crowd workers in a crowdsourcing platform. Second, we perform a CF mechanism based on the nearest neighbors to integrate the item neighbors obtained from the workers into new item recommendations.

By utilizing crowdsourcing, we can effectively find item neighbors for new items, thereby increasing the accuracy of the new item recommendation. Although we proposed several ideas to effectively utilize crowdsourcing (*e.g.*, the design of two-step tasks, worker management, task assignment), our framework has inherent limitations such as time latency, cost, and quality of workers in using crowdsourcing. These limitations have been extensively studied in existing crowdsourcing problems (Garcia-Molina et al., 2016; Guo et al., 2014; Li et al., 2016). As future work, we leave various optimizations to overcome inherent crowdsourcing problems for item recommendation.

4. Experiments

In this section, we evaluate the effectiveness of the CrowdStart framework. We first present an experimental setup and evaluation protocols for user study and simulation tests. We then report experimental results by comparing the CrowdStart framework against existing content-based and hybrid algorithms using textual information.

4.1. Experimental setup

Dataset. We used the latest MovieLens dataset, which is widely used for evaluating recommender systems (<https://grouplens.org/>). This dataset includes 100,000 ratings of 700 users and 9000 movies released from 1995 to 2016. We used basic details from the MovieLens dataset such as genres, directors, and tags. Besides, we collected movie plots and poster images for each movie from Wikipedia and IMDb (<http://www.imdb.com/>).

To evaluate the new item recommendations of the CrowdStart framework, we chose 20 movies as target items. They can be categorized into two groups:

- New movies:** These movies had not been released at the time of our evaluation. This group contains the following movies: Fantastic Beasts and Where to find them, Arrival, Allied, La La Land, Rogue one, Wonder Woman, Rough Night, The Book of Henry, Cars 3, and All Eyez on Me.

Select relevant movies (Click to collapse)

Please select the movie by reading the guidelines and movie information given below.

Guidelines:

- Select more relevant movies (5 or more) out of 21.
- Write the reason why these movies are relevant
- The example of the movie relevance: the similarity of synopsis, the similarity of genres.
- Please do not select 'empty' cell.

Movie informations:

- Movie trailer: [Movie trailer url](#)

- Title: Wonder Woman

- Genres: Action,Adventure,Fantasy

- Director: Patty Jenkins


- Stars: Gal Gadot,David Thewlis,Robin Wright

- Plot summary:


- Before she was Wonder Woman, she was Diana, princess of the Amazons, trained to be an unconquerable warrior. Raised on a sheltered island paradise, when an American pilot crashes on their shores and tells of a massive conflict raging in the outside world, Diana leaves her home, convinced she can stop the threat. Fighting alongside man in a war to end all wars, Diana will discover her full powers and her true destiny.
- Before she was Wonder Woman she was Diana, princess of the Amazons, trained warrior. When a pilot crashes and tells of conflict in the outside world, she leaves home to fight a war to end all wars, discovering her full powers and true destiny.

More information: [IMDB information url](#)


* If you don't know a movie given below, you should search that movie in : [IMDB](#)




Batman v Superman: Dawn of Justice




War horse




Superman



Avatar



Mad Yax



Underworld

Fig. 4. The user interface of the voting task with the following parts–(A) showing a guideline with examples, (B) providing the details of a new movie, and (C) clicking relevant movies.

- **Existing movies:** These movies had been released, and their ratings existed in the MovieLens dataset at the time of our evaluation. This group contains the following movies: William Shakespeare's Romeo + Juliet, Glory, Boogie Nights, You've Got Mail, The Others, Black Hawk Down, I Robot, Walk the Line, 300, and Superbad.

For each movie in the two groups, the crowd-based neighbors (obtained in steps 1 and 2 of the CrowdStart framework) are evaluated by a user study. We then perform a simulation test for recommending the new item (which is done in step 3 of the CrowdStart framework).

Crowd-based neighbors. To validate our CrowdStart framework, we construct top- N crowd-based neighbors for each target movie. In our experiments, we choose $N=30$ that is the average number of ratings for users in the MovieLens dataset. In the CrowdStart framework, steps 1 and 2 for constructing the crowd-based neighbors are built upon MTurk. As mentioned in Section 3.2, we control various features for effective worker management and task assignment in crowdsourcing tasks.

We report detailed settings and statistics on the features. For all tasks, we commonly use the qualification requirement of a HIT approval rate $\geq 90\%$ and the number of HIT's approved ≥ 500 on MTurk. For step 1, 400 workers participate in generating relevant movies for 20 target movies (20 workers per target movie). In this process, we generate 36 candidate movies on average (min. = 27, and max. = 50). Also, we manually eliminate the candidate movies

obtained by untrustworthy workers and collect additional candidate movies by using existing algorithms. To do this, we can use any algorithms that deal with the cold-start item problem mentioned in Section 2. In this paper, we simulate our framework with two algorithms that utilize textual information, CNN and LDA. After completing the generation task, we have finally 420 candidate movies (21 candidate movies per target movie) obtained by crowd workers and existing algorithms.

Then, for step 2, 600 workers (different from workers of step 1) participate in voting on the most relevant movies among the candidate movies for 20 target movies (30 workers per target movie). After that, we manually eliminate the candidate movies obtained by untrustworthy and spam workers and select the top-30 movies with the highest votes as the final crowd-based neighbors. For each step, we paid \$0.2 and \$0.1 per movie to workers, respectively. Also, the average time per assignment for each step is 5 min and 1.5 min, respectively. Also, we found that only 2.2% of workers ($=22/1000$) among all workers were filtered out as untrustworthy or spam workers. Table 1 summarizes the statistics for the crowdsourcing tasks.

Competing algorithms. For comparative experiments, we construct four sets of top- N ($N = 30$) machine-based neighbors for each target movie based on existing algorithms. To do this, we employ four existing algorithms using textual information, LDA, CNN, CTR, and ConvMF. Specifically, they first learn textual information of all movies using LDA (LDA and CTR) or CNN(CNN and Con-

Table 1
Statistics for our crowdsourcing tasks.

	# of workers	Cost	Time
Generating task	400 (20 per movie)	\$80 (\$0.2 per movie)	4 days (Avg. time per assignment: 5 min)
Voting task	600 (30 per movie)	\$60 (\$0.1 per movie)	1 day (Avg. time per assignment: 1.5 min)

vMF). Then, they calculate the similarities between a target movie and other movies based on learned results and identify the top-30 movies with the highest similarities to the target movie. Lastly, the top-30 movies are considered as *content-based neighbors* for each target movie. In that case, CTR and ConvMF use not only text information but also a user-item rating matrix. We thus call neighbors obtained by them as *rating-based neighbors with additional information*.

Note that the five sets may contain the same movies. Recall that a set of neighbors obtained by CrowdStart may contain movies existed in those obtained by CNN and LDA.

4.2. Evaluation protocols

We validate the CrowdStart framework from two directions: (1) the quality of crowd-based neighbors and (2) the accuracy of new item recommendations using the neighbors. The evaluation is conducted in two ways: a user study and simulation tests.

Evaluation of crowd-based neighbors. We perform a user study using crowdsourcing. We adopt two metrics to compare the crowd-based neighbors against content-based and rating-based neighbors obtained by LDA, CNN, CTR, and ConvMF:

- **Reliability:** How reliable are item neighbors in terms of the relevance to the target movie?
- **Diversity:** How diverse are item neighbors from the target movie in terms of the characteristics of item neighbors?

For a fair user study, we made five types of neighbors (i.e., a crowd-based, two content-based, and two rating-based) as blind item groups. Here, the five groups of neighbors may contain the same items each other.⁶ Then, we asked 100 crowd workers (different from workers of steps 1 and 2) to answer the following two questions: (1) Which group is more appropriate for each metric? (i.e., ranking five groups relatively) and (2) how satisfactory is each group for each metric? (i.e., scoring each group with a five-point Likert scale from “strongly disagree” to “strongly agree”).

Evaluation of new item recommendations. We conduct a simulation test for new item recommendation. Given a target item t , let U_t be a set of users who rated target item t . For use as ground truth in the simulation test, we deliberately remove the ratings of U_t for t . Furthermore, users with many ratings are more likely to find themselves similar to the virtual user than users with a few ratings, so the results of new item recommendations may be biased to heavy-rating users. Therefore, we exclude those users with too many ratings. In the MovieLens dataset, we found that the top-30 users (i.e., about 5%) have 29,990 ratings (i.e., about 30%). So, we filter the top-30 users out as heavy-rating users.

Given each set of item neighbors for target item t obtained by LDA, CNN, CTR, ConvMF, and CrowdStart, the simulation test is performed as follows. First, we append a new virtual user v who has rated all the item neighbors and target item t in R . Then, we identify a set K of top- k users who have similar tastes as virtual

Table 2
Proportion and number of votes of movies collected by each method.

	Crowd workers	CNN	LDA
Proportion (%)	50.6	17.1	30.3
Avg. # of votes	3.9	1.2	2.2

user v . To measure the similarity between users, we used two similarity measures: the *number of co-rated items* (Eq. (1)) and the *binary cosine similarity* (Eq. (2)).

$$\text{sim}(u, v) = |I_u \cap I_v| \quad (1)$$

$$\text{sim}(u, v) = \frac{I_u \cdot I_v}{\sqrt{|I_u| |I_v|}} \quad (2)$$

Here, I_u and I_v are two sets of items that are rated by user u and virtual user v , respectively. Lastly, we recommend target item t to k users in K .

Note that, because ten new movies (out of 20 target movies) have no ratings in the MovieLens dataset, we cannot generate the ground truth for them in the simulation test. We thus evaluate new item recommendation only using the ten existing movies in the MovieLens dataset. To measure the accuracy of new item recommendations, we adopt two well-known measures: *precision@k* and *recall@k*. In the experimental results, we will report the average of precision and recall for ten movies.

4.3. Experimental results

To evaluate the CrowdStart framework, we attempt to answer the following questions:

- Q1: Can crowd workers generate more neighbors relevant to new items than machine-only algorithms?
- Q2: Are crowd-based neighbors more reliable and more diverse than content-based and rating-based neighbors?
- Q3: Is the CrowdStart framework using crowd-based neighbors useful for new item recommendation?
- Q4: Does the CrowdStart framework provide reasonable explanations for recommendation results?

4.3.1. The effectiveness of crowd workers (Q1)

The CrowdStart framework eventually generates crowd-based neighbors by selecting the top-30 movies with the most votes among which are candidate movies collected by both crowd workers and existing algorithms. In this paper, we employ two existing algorithms that use textual information, CNN and LDA. To answer (Q1), we analyzed the distribution of movies finally selected as crowd-based neighbors among candidate movies collected by crowd workers, CNN, and LDA.

Table 2 shows (1) the percentage of candidate movies collected by crowd workers, CNN, and LDA among all crowd-based neighbors for 20 target movies and (2) the number of votes received by crowd workers of step 2 for candidate movies in each method. The number of votes in Table 2 indicates the average number of votes in all crowd-based neighbors for 20 target movies. Note that the crowd workers who perform the generating task and the voting

⁶ Note that, in generating task, CrowdStart collects candidate items by both crowd workers and existing algorithms, CNN and LDA. Also, CTR and ConvMF learn textual information of items using LDA and CNN, respectively. However, we found that most of the neighbors in each set do not overlap with each other. The neighbors included in each set can be found at <https://goo.gl/1iXTUE>.

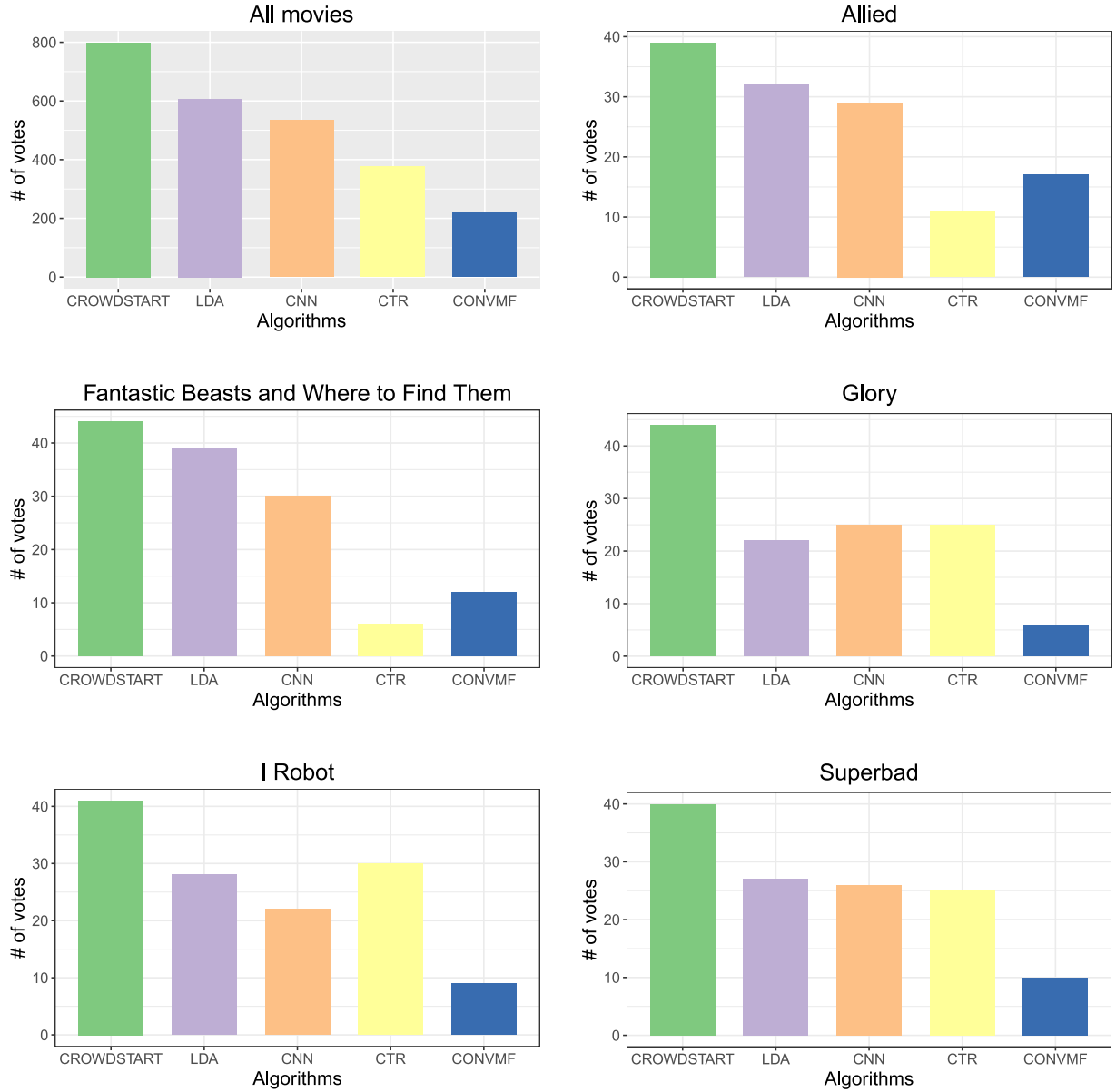


Fig. 5. Comparison of the number of votes about reliability for CrowdStart, LDA, CNN, CTR, and ConvMF: The top left one is the total number of votes for all movies, while the others are user study results of 5 movies 'Allied', 'Fantastic Beasts and Where to Find Them', 'Glory', 'I Robot', and 'Superbad'.

task are different. We found that the movies collected by crowd workers have more votes than CNN and LDA. It is shown that crowd workers are more effective than machine-only algorithms in finding neighbors relevant to new items in terms of user's preferences.

4.3.2. Quality of crowd-based neighbors (Q2)

For (Q2), we compared the quality of five sets of neighbors obtained by CrowdStart, LDA, CNN, CTR, and ConvMF through a user study. As mentioned in Section 4.2, we asked 100 crowd workers to evaluate the reliability and diversity of crowd-based neighbors using qualitative and quantitative questions (i.e., ranking of groups (Q2-1) and Likert-scale score (Q2-2)).

For (Q2-1), Figs. 5 and 6 show the numbers of votes received by crowd workers for each algorithm. The x-axis represents each algorithm, and the y-axis represents the number of votes for each algorithm ranked 1st in terms of reliability (Fig. 5) and diversity (Fig. 6). The top-left graph shows the average number of votes

for each algorithm for all target movies, and the remainder shows those for (randomly selected) five movies out of all the target movies.

In Fig. 5, the CrowdStart framework is shown to consistently and significantly outperform LDA, CNN, CTR, and ConvMF. In particular, the total number of votes for CrowdStart is higher than LDA, CNN, CTR, and ConvMF by 31%, 49%, 111%, and 259%, respectively. Similarly, for each selected movie, the number of votes for CrowdStart is always the highest. This result indicates that the CrowdStart framework finds the most reliable neighbors to target movies than the other algorithms. Next, Fig. 6 shows the diversity result. Similar to Fig. 5, it is found that the CrowdStart framework outperforms LDA, CNN, CTR, and ConvMF. Specifically, the total number of votes for CrowdStart is higher than LDA, CNN, CTR, and ConvMF by 5%, 14%, 15%, and 21%, respectively.

For (Q2-2), Figs. 7 and 8 show the results of evaluating the neighbors obtained by each algorithm based on the Likert-scale in terms of reliability and diversity. The percentage on both ends

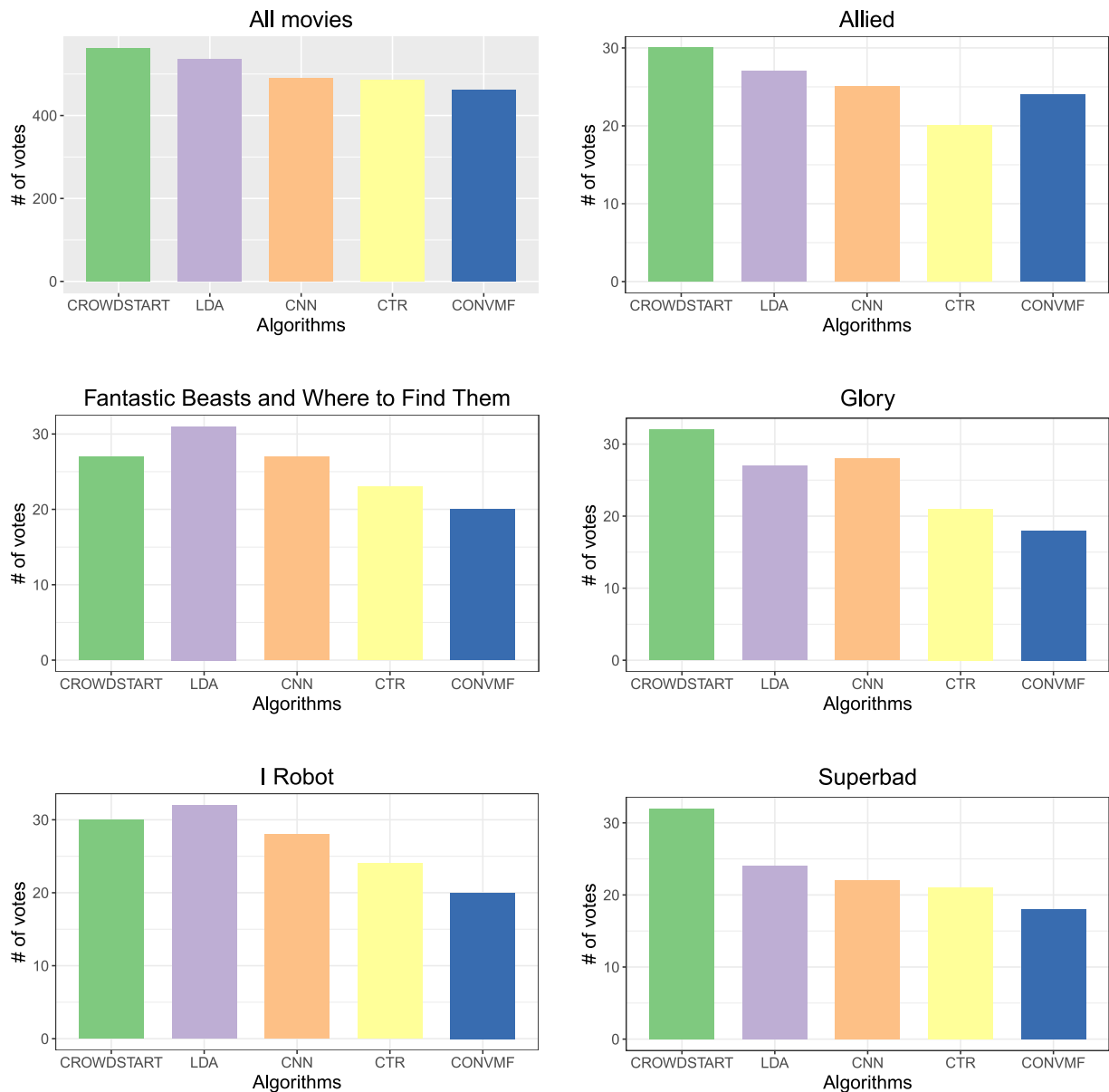


Fig. 6. Comparison of the number of votes about diversity for CrowdStart, LDA, CNN, CTR, and ConvMF: The top left one is the total number of votes for all movies, while the others are user study results of 5 movies ‘Allied’, ‘Fantastic Beasts and Where to Find Them’, ‘Glory’, ‘I Robot’, and ‘Superbad’.

means the proportion of each algorithm having received “disagree” and “agree” votes by crowd workers, and the middle percentage is the proportion of their neutral votes, respectively.

In Fig. 7, on average 68% of users agreed that the crowd-based neighbors obtained from the CrowdStart framework were reliable. Furthermore, the proportion of “disagree” votes for the crowd-based neighbors was the least percentage at 6%, and the proportion of “strong agree” votes was much higher than the other algorithms. Compared to content-based and rating-based neighbors from LDA, CNN, CTR, and ConvMF, the results of CrowdStart show that users are more satisfied with its reliability.

In Fig. 8, we compare the diversity for five types of neighbors. Note that, in this paper, we aim to provide relevant and diverse neighbors to new items. However, it is known that irrelevant movies highly influence the diversity of neighbors. In other words, if the neighbors include more movies not related to the target movie, the diversity of the neighbors can be judged to be higher. Thus, the diversity results can be biased towards machine-

only algorithms involving many irrelevant neighbors compared to CrowdStart (see Fig. 7). Nevertheless, the evaluation for diversity of CrowdStart is best compared to other algorithms. Surprisingly, even the proportion of “strong agree” votes is the highest. It indicates that our crowd-based neighbors not only have significantly more reliability than the machine-based neighbors but also provides higher diversity than or comparable to them.

4.3.3. The accuracy of new item recommendations (Q3)

For (Q3), we conducted simulation tests to demonstrate the accuracy of item neighbors for new item recommendations. In the test, we first leverage a virtual user with a preference for each algorithm’s neighbors. Then, we validate that crowd-based neighbors are more helpful in finding users who are likely to prefer the new movie than content-based and rating-based neighbors. To do this, we compare how many users out of the top- k users who prefer the neighbors obtained from each algorithm also prefer the new movie.

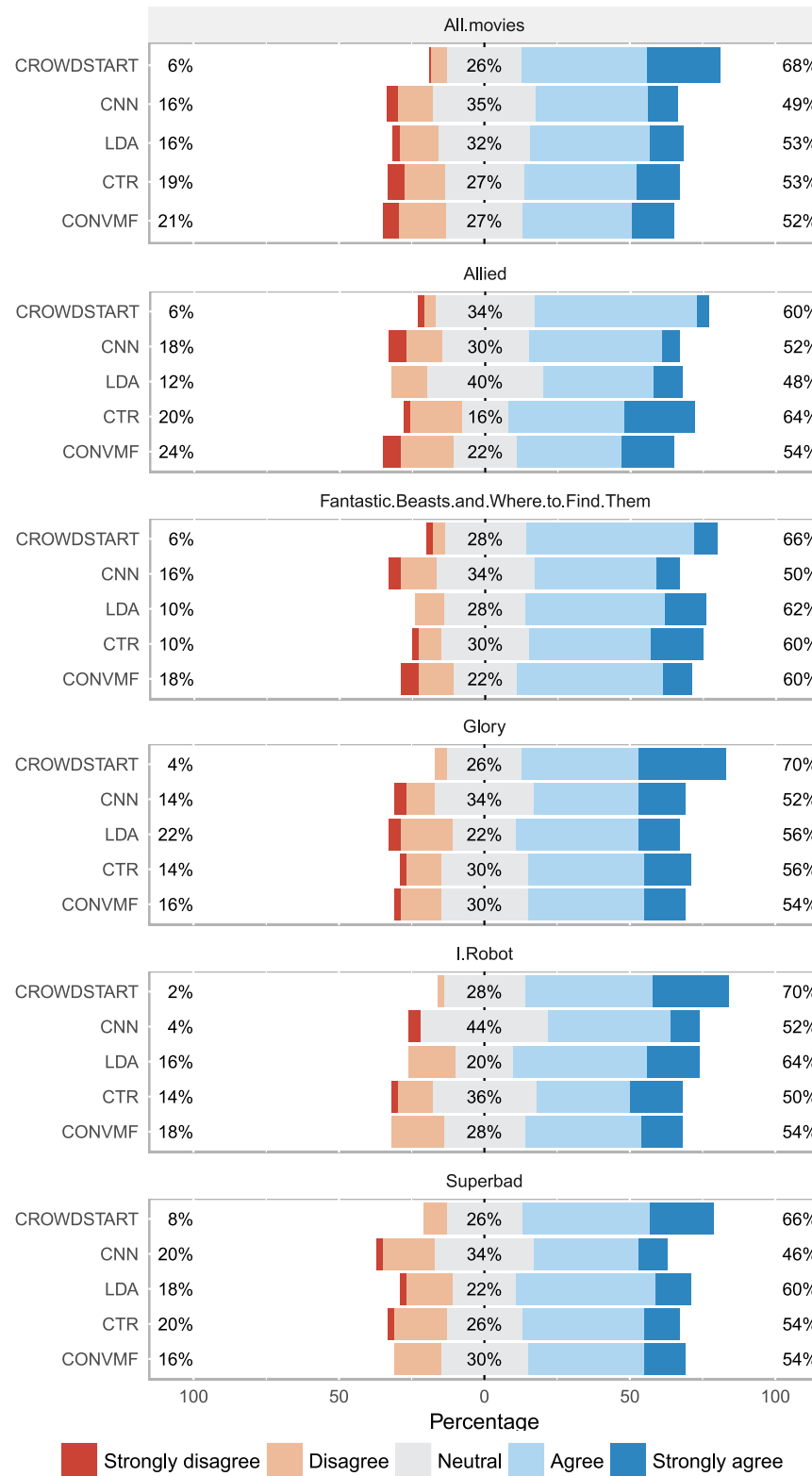


Fig. 7. Comparison of the proportion of Likert-scale scores about reliability for CrowdStart, LDA, CNN, CTR, and ConvMF: The first one is the average proportion of Likert-scale scores for all movies. The others are individual results of 5 movies "Allied," "Fantastic Beasts and Where to Find Them," "Glory," "I Robot," and "Superbad."

Fig. 9 shows the accuracy of the top- k recommendations using each algorithm's neighbors based on two similarity measures, the number of co-rated items and binary cosine similarity, respectively. The x -axis represents k (from 10 to 100) of the top- k movies, and the y -axis represents the precision and recall.

Fig. 9(a) shows the result of recommendation based on the number of co-rated items. This result indicates that CrowdStart shows comparable with CNN and significantly better (in all k except 40) accuracy than other algorithms. However, the similarity measure (i.e., the number of co-rated items) of Fig. 9(a) does not

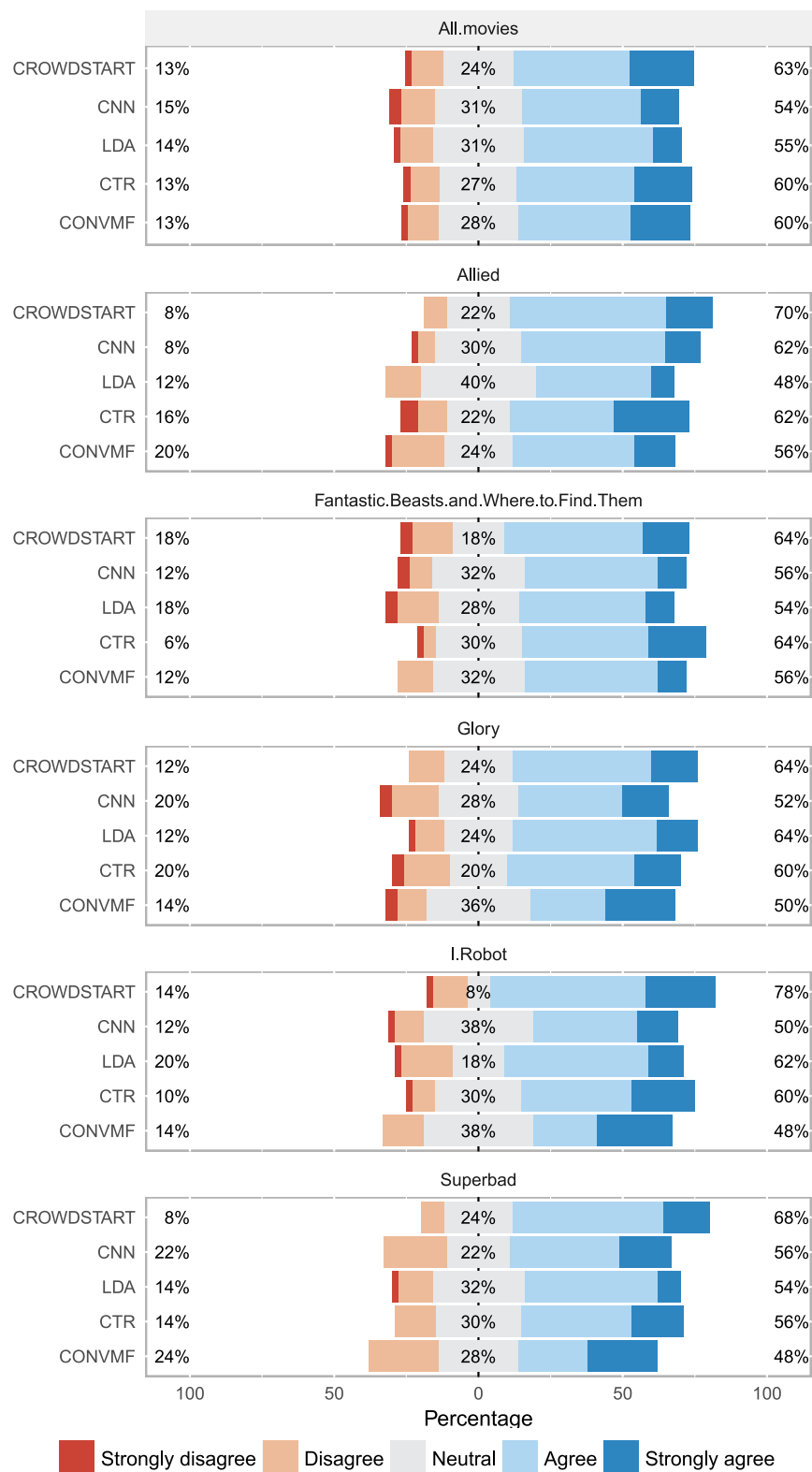


Fig. 8. Comparison of the proportion of Likert-scale scores about diversity for CrowdStart, LDA, CNN, CTR, and ConvMF: The first one is the average proportion of Likert-scale scores for all movies. The others are individual results of 5 movies "Allied," "Fantastic Beasts and Where to Find Them," "Glory," "I Robot," and "Superbad."

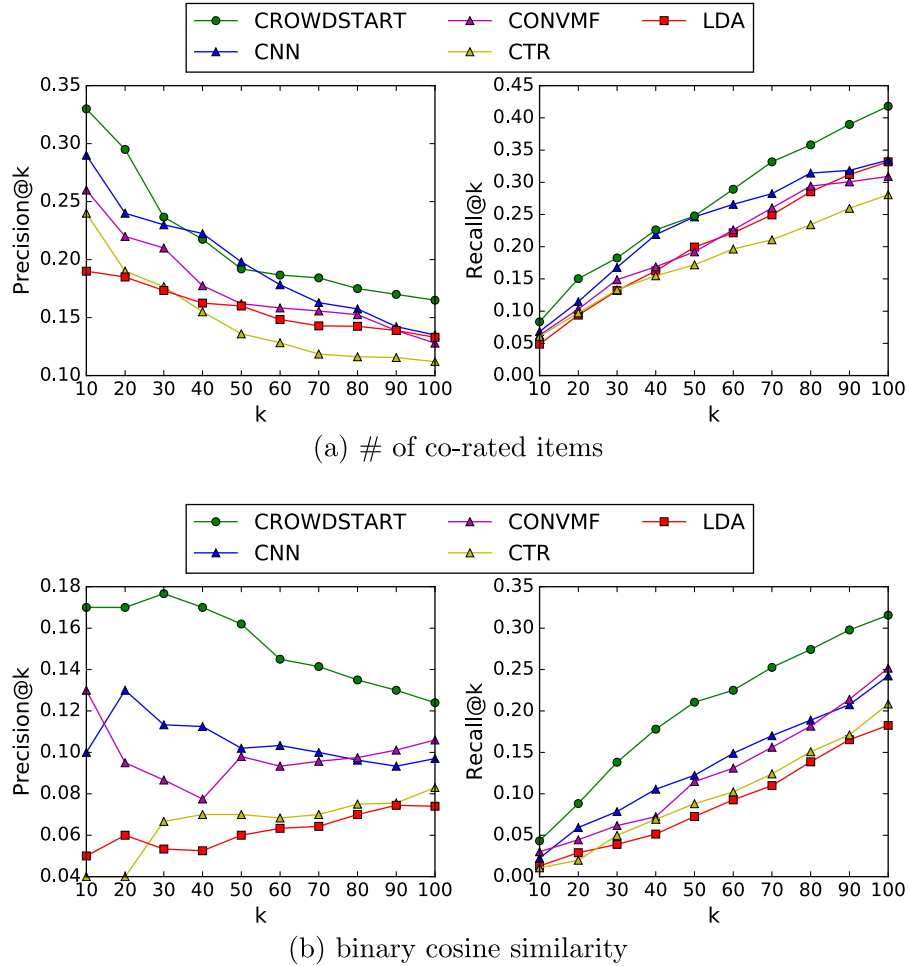


Fig. 9. Accuracy of recommendation using each neighbor obtained by LDA, CNN, CTR, ConvMF, and CrowdStart.

consider the number of ratings that the users have. Because users with a relatively high number of ratings are likely to have a rating for the new item, this measure can still be biased by these users even though the heavy-rating users have been removed in advance. That is, recommending items to these users is a trivial and easy task.

In Fig. 9(b), we thus verify the result of recommendation based on binary on binary cosine similarity, which considers the number of ratings of users. In all parameters k , CrowdStart consistently outperforms other algorithms. Specifically, the precision of CrowdStart for the top-30 recommendations is shown to be higher by 56% than that of CNN. Similarly, the recall of CrowdStart is always higher than that of other algorithms by up to 75% (in top@50). We confirmed that CrowdStart effectively recommends new items to users with a relatively low number of ratings. In summary, these results show that the crowd-based neighbors relevant to a new item are the most effective in finding users who prefer the new item compared to other neighbors.

4.3.4. Explanations for the recommendation results (Q4)

As mentioned in Section 3, crowd workers fill in the reasons why items they selected/voted on are similar to new items. Using various reasons collected from different crowd workers, we can provide reasonable explanations for the recommendation results of the CrowdStart framework. That is, in addition to recommending new items to some users, we can also provide reasons why the new items might be preferred by the user.

Table 3 lists three target movies “Black Hawk Down,” “Arrival,” and “Cars 3,” and four movies selected as crowd-based neighbors for each target movie. It also includes the explanations by the crowd workers who selected each movie along with descriptions of each movie. For example, the explanations for the neighbors of “Black Hawk Down” can be summarized as follows: (1) they are similar genres and (2) the characters experience similar situations. It was also observed that the crowd-based neighbors of “Arrival” and “Cars3” are similar.

When recommending a new movie to a user, therefore, we not only can provide other movies (i.e., crowd-based neighbors) related to the movie but also can provide reasons why those movies are relevant. This allows the user to get a better understanding of why the new movie was recommended.

4.3.5. Summary for experimental results

To conclude, implications based on the experimental results are summarized as follows. First, we shown that existing machine-only algorithms fail to capture the comprehensive understanding of underlying similarities among items. This result indicates that machine-only solutions are still limited for considering reliable and diverse relationships to find latent similarities between items. Second, by utilizing human knowledge obtained from crowd workers, we can capture effectively latent similarities between items, and then find relevant, diverse, and explainable neighbors for new items. In other words, we shown that by integrating crowdsourcing into one part of machine-only algorithms, they can effectively address the cold-start problem that is difficult to solve by them-

Table 3

Three target new movies “Black Hawk Down”, “Arrival”, and “Cars 3”, and four crowd-based neighbors with explanations for each target movie.

(a) New Movies	
Title (Genres)	Plots for new movies
Black Hawk Down (Drama, War)	160 elite U.S. soldiers drop into Somalia to capture two top lieutenants of a renegade warlord and find themselves in a desperate battle with a large force of heavily-armed Somalis.
Arrival (Mystery, Sci-Fi)	A linguist is recruited by the military to communicate with alien lifeforms after twelve mysterious spacecrafts land around the world.
Cars 3 (Animation, Adventure)	Lightning McQueen sets out to prove to a new generation of racers that he's still the best race car in the world.
(b-1) Crowd-based Neighbors of “Black Hawk Down”	
Title (Genres)	Explanations for the neighbors
Saving Private Ryan (Action, Drama)	“I would also consider this movie to be in the genres of history drama and war also.” “ The time period is in WWII were a group of soldiers have to finds Private Ryan , and make sure he gets home safely because he is the last to carry on is name because, all his brothers died in the war.”
Pearl Harbor (Action, Drama)	“Pearl Harbor is also a history, war, drama .” “Two friends who are pilots in WWII together and also fall in love with the same girl.”
Behind Enemy Lines (Drama, Thriller)	“ Similar plot in which a plane crashes behind enemy lines and how the pilot struggles to reach safe territory.”
Rescue Dawn (Action, Adventure)	“ A U.S. fighter pilot's epic struggle of survival after being shot down on a mission over Laos during the Vietnam War.”
(b-2) Crowd-based Neighbors of “Arrival”	
Title (Genres)	Explanations for the neighbors
Alien vs predator (Horror, Sci-Fi)	“This movie is about a group of explores finding an underground ancient castle which is infested with creatures . These creatures come to earth and kill a bunch of people.”
Independence Day (Adventure, Sci-Fi)	“This movie is Sci-fi as well and features aliens .” “Because it is a story about extraterrestrials coming to Earth and humanity needs to find a way to save mankind.”
Contact (Mystery, Sci-Fi)	“Something alien is discovered and a team has to find out what it is and where it comes from.” “Another film in which humans try to communicate with extraterrestrials .”
Skyline (Action, Sci-Fi)	“It is a sci-fi movie involving alien invasion , the leading characters bravely fought with the situation.”
(b-3) Crowd-based Neighbors of “Car 3”	
Title (Genres)	Explanations for the neighbors
Cars (Animation, Comedy)	“Cars 3 is the third movie in the “Cars” franchise . This is the film where it all began and Lightning McQueen's adventure began , making it an important watch.”
Cars 2 (Animation, Adventure)	“This movie is another animated children's movie by Pixar Animation. Cars 2 is the second movie in the Cars series of three movies . It has the same main characters as Cars 3.”
Turbo (Animation, Adventure)	“Like Cars 3, this is an animated movie about one character's dream of racing . This is similar because it is an animated children's movie , and the plot is heavily focused around a big race .”
Planes (Animation, Adventure)	“This movie is similar to Cars because it brings inanimate objects (Planes) to watch and it's a fun, family-friendly movie that kids will love .”

selves. We believe that human wisdom will be effective in solving various complex problems other than the cold-start problem.

5. Conclusion and future work

To alleviate the cold-start problem, we proposed a novel *crowd-enabled* framework, called CrowdStart, that utilizes the wisdom of crowds via crowdsourcing. Our intuition behind the CrowdStart framework is based on the conventional expert systems, where the knowledge of domain experts is useful for solving complex problems that difficult to solve with machine-only algorithms. Unlike existing expert systems, we take advantage of crowdsourcing to employ large-scale human workers in a crowdsourcing platform.

To incorporate human knowledge to new item recommendation, we designed an integrated human-machine approach. The CrowdStart framework consists of three steps: (1) generating diverse candidate neighbors for the new item, (2) voting on the most reliable items for the new item, and (3) incorporating a set of items selected by crowd workers into new item recommendation via CF mechanism. To evaluate the effectiveness of the CrowdStart framework, we lastly conducted extensive experiments including both a user study using crowdsourcing and simulated tests. The experimental results show that the crowd workers provide *relevant, diverse, reliable, and explainable crowd-based neighbors* for the new item as well as the crowd-based neighbors are helpful for new item recommendation.

As our future work, we will address several limitations for our framework in terms of time latency, cost, and quality in hiring and managing crowd workers. That is, we plan to conduct studies to design crowdsourcing tasks with the goal of reducing time latency,

being cost-effective, and controlling workers' quality. It is also possible to combine the existing two-step crowdsourcing tasks to a single task by selectively hiring only those workers with expertise (i.e., workers with high quality). Also, we plan to investigate various ways of building a general recommendation framework that can consider both new items and existing items within a single mechanism.

Conflict of interest

The authors declare that there is no conflict of interests regarding the publication of this article.

Credit authorship contribution statement

Dong-Gyun Hong: Data curation, Formal analysis, Methodology, Writing - original draft. **Yeon-Chang Lee:** Data curation, Formal analysis, Methodology, Writing - original draft. **Jongwuk Lee:** Data curation, Formal analysis, Methodology, Writing - original draft. **Sang-Wook Kim:** Data curation, Formal analysis, Methodology, Writing - original draft.

Acknowledgements

This work was supported by the [National Research Foundation of Korea \(NRF\)](#) grant funded by the Korea government (MSIT; Ministry of Science and ICT) ([NRF-2017R1A2B3004581](#), [2018R1A2B6009135](#), and [2018R1A5A7059549](#)) and Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the

Ministry of Science and ICT (NRF-2017M3C4A7083678). This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (no. 2019-0-00421, AI Graduate School Support Program).

References

- Adomavicius, G., & Tuzhilin, A. (2005). 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), 734–749.
- Chang, J. C., Amershi, S., & Kamar, E. (2017). Revolt: Collaborative crowdsourcing for labeling machine learning datasets. In *ACM international conference on human factors in computing systems (ACM CHI)* (pp. 2334–2346).
- Chang, S., Harper, F. M., He, L., & Terveen, L. G. (2016). Crowdlens: Experimenting with crowd-powered recommendation and explanation. In *AAAI international conference on web and social media (ICWSM)* (pp. 52–61).
- Chang, S., Harper, F. M., & Terveen, L. (2016). Crowd-based personalized natural language explanations for recommendations. In *ACM international conference on recommender systems (ACM RecSys)* (pp. 175–182).
- Felfernig, A., Haas, S., Ninaus, G., Schwarz, M., Ulz, T., Stettinger, M., et al. (2014). Recturk: Constraint-based recommendation based on human computation. *ACM RecSys 2014 CrowdRec workshop*.
- Felfernig, A., Ulz, T., Haas, S., Schwarz, M., Reiterer, S., & Stettinger, M. (2015). Peopleviews: Human computation for constraint-based recommendation. *ACM RecSys 2015 CrowdRec workshop*.
- Garcia-Molina, H., Joglekar, M., Marcus, A., Parameswaran, A. G., & Verroios, V. (2016). Challenges in data crowdsourcing. *IEEE Transactions on Knowledge and Data Engineering*, 28(4), 901–911.
- Ghosh, S., Sharma, N. K., Benevenuto, F., Ganguly, N., & Gummadi, P. K. (2012). Cognos: Crowdsourcing search for topic experts in microblogs. In *ACM international conference on research and development in information retrieval (ACM SIGIR)* (pp. 575–590).
- Guo, G., Zhang, J., & Yorke-Smith, N. (2015). Trustsvd: Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings. In *AAAI international conference on artificial intelligence (AAAI)* (pp. 123–129).
- Guo, G., Zhang, J., Zhu, F., & Wang, X. (2017). Factored similarity models with social trust for top-n item recommendation. *Knowledge-Based Systems*, 122, 17–25.
- Guo, X., Wang, H., Song, Y., & Hong, G. (2014). Brief survey of crowdsourcing for data mining. *Expert Systems with Applications*, 41(17), 7987–7994.
- He, R., Lin, C., Wang, J., & J. M. (2016). Sherlock: sparse hierarchical embeddings for visually-aware one-class collaborative filtering. *International Joint Conference on Artificial Intelligence (IJCAI)*, 3740–3746.
- He, R., & McAuley, J. (2016a). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *International conference on world wide web (WWW)* (pp. 507–517).
- He, R., & McAuley, J. (2016b). VBPR: Visual bayesian personalized ranking from implicit feedback. In *AAAI international conference on artificial intelligence (AAAI)* (pp. 144–150).
- Ipeirotis, P. G., & Gabrilovich, E. (2014). Quizz: Targeted crowdsourcing with a billion (potential) users. In *International conference on world wide web conference (WWW)* (pp. 143–154).
- Jackson, P. (1986). *Introduction to Expert Systems* (1st ed.). Addison-Wesley.
- Jamali, M., & Ester, M. (2009). Trustwalker: A random walk model for combining trust-based and item-based recommendation. In *ACM international conference on knowledge discovery and data mining (ACM KDD)* (pp. 397–406).
- Jamali, M., & Ester, M. (2010). A matrix factorization technique with trust propagation for recommendation in social networks. In *ACM international conference on recommender systems (ACM RecSys)* (pp. 135–142).
- Kim, D., Park, C., Oh, J., Lee, S., & Yu, H. (2016). Convolutional matrix factorization for document context-aware recommendation. In *ACM international conference on recommender systems (ACM RecSys)* (pp. 233–240).
- Lee, J., Jang, M., Lee, D., Hwang, W.-S., Hong, J., & Kim, S.-W. (2013). Alleviating the sparsity in collaborative filtering using crowdsourcing. *ACM RecSys 2013 CrowdRec workshop*.
- Li, G., Wang, J., Zheng, Y., & Franklin, M. J. (2016). Crowdsourced data management: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 28(9), 2296–2319.
- Li, S., Kawale, J., & Fu, Y. (2015). Deep collaborative filtering via marginalized denoising auto-encoder. In *ACM international conference on information and knowledge management (ACM CIKM)* (pp. 811–820).
- Lika, B., Kolomvatsos, K., & Hadjiethymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4), 2065–2073.
- Lin, X., Zhang, M., Zhang, Y., Liu, Y., & Ma, S. (2017). Learning and transferring social and item visibilities for personalized recommendation. In *ACM international conference on information and knowledge management (ACM CIKM)* (pp. 337–346).
- Liu, Q., Wu, S., & Wang, L. (2017). Deepstyle: Learning user preferences for visual recommendation. In *ACM international conference on research and development in information retrieval (ACM SIGIR)* (pp. 841–844).
- Ma, H., Lyu, M. R., & King, I. (2009). Learning to recommend with trust and distrust relationships. In *ACM international conference on recommender systems (ACM RecSys)* (pp. 189–196).
- Ma, H., Yang, H., Lyu, M. R., & King, I. (2008). SoRec: Social recommendation using probabilistic matrix factorization. In *ACM international conference on information and knowledge management (ACM CIKM)* (pp. 931–940).
- Ma, H., Zhou, D., Liu, C., Lyu, M. R., & King, I. (2011). Recommender systems with social regularization. In *ACM international conference on web search and data mining (ACM WSDM)* (pp. 287–296).
- Mohit, K., Singh, B., & Gurpreet, S. (2016). K-means demographic based crowd aware movie recommendation system. *Indian Journal of Science and Technology*, 9(23), 1–4.
- Moradi, P., & Ahmadian, S. (2015). A reliability-based recommendation method to improve trust-aware recommender systems. *Expert Systems with Applications*, 42(21), 7386–7398.
- Organisciak, P., Teevan, J., Dumais, S., Miller, R. C., & Kalai, A. T. (2014). A crowd of your own: Crowdsourcing for on-demand personalization. In *AAAI international conference on human computation and crowdsourcing (HCOMP)* (pp. 211–218).
- Organisciak, P., Teevan, J., Dumais, S., Miller, R. C., & Kalai, A. T. (2015). Matching and grokking: Approaches to personalized crowdsourcing. In *International joint conference on artificial intelligence (IJCAI)* (pp. 4296–4302).
- Park, S.-T., & Chu, W. (2009). Pairwise preference regression for cold-start recommendation. In *ACM international conference on recommender systems (ACM RecSys)* (pp. 21–28).
- Rafailidis, D., & Crestani, F. (2017). Learning to rank with trust and distrust in recommender systems. In *ACM international conference on recommender systems (ACM RecSys)* (pp. 5–13).
- (2015). In F. Ricci, L. Rokach, & B. Shapira (Eds.), *Recommender systems handbook*. Springer.
- Said, A., Larson, M., Tikk, D., Cremonesi, P., Karatzoglou, A., Hopfgartner, F., et al. (2014). User-item reciprocity in recommender systems: Incentivizing the crowd. *Posters, demos, late-breaking results and workshop proceedings of the conference on user modeling, adaptation, and personalization (UMAP)*.
- Shi, L., Wu, B., Zheng, J., Shi, C., & Li, M. (2017). DSBPR: Dual similarity Bayesian personalized ranking. In *Pacific-Asia conference on knowledge discovery and data mining (PAKDD)* (pp. 266–277).
- Smyth, B., Raftar, R., & Banks, S. (2016). Harnessing crowdsourced recommendation preference data from casual gameplay. In *ACM international conference on user modeling adaptation and personalization (ACM UMAP)* (pp. 95–104).
- Tang, J., Aggarwal, C., & Liu, H. (2016). Recommendations in signed social networks. In *International conference on world wide web (WWW)* (pp. 31–40).
- Wang, C., & Blei, D. M. (2011). Collaborative topic modeling for recommending scientific articles. In *ACM international conference on knowledge discovery and data mining (ACM KDD)* (pp. 448–456).
- Wang, H., Wang, N., & Yeung, D.-Y. (2015). Collaborative deep learning for recommender systems. In *ACM international conference on knowledge discovery and data mining (ACM KDD)* (pp. 1235–1244).
- Wang, S., Wang, Y., Tang, J., Shu, K., Ranganath, S., & Liu, H. (2017). What your images reveal: Exploiting visual contents for point-of-interest recommendation. In *International conference on world wide web (WWW)* (pp. 391–400).
- Wang, Y. (2002). Predicting stock price using fuzzy grey prediction system. *Expert Systems with Applications*, 22(1), 33–38.
- Wei, J., He, J., Chen, K., Zhou, Y., & Tang, Z. (2017). Collaborative filtering and deep learning based recommendation system for cold start items. *Expert Systems with Applications*, 69, 29–39.
- Wu, H., Zhang, Z., Yue, K., Zhang, B., He, J., & Sun, L. (2018). Dual-regularized matrix factorization with deep neural networks for recommender systems. *Knowledge-Based Systems*, 145, 46–58.
- Wu, Y., DuBois, C., Zheng, A. X., & Ester, M. (2016). Collaborative denoising auto-encoders for top-n recommender systems. In *ACM international conference on web search and data mining (ACM WSDM)* (pp. 153–162).
- Yao, W., He, J., Wang, H., Zhang, Y., & Cao, J. (2015). Collaborative topic ranking: Leveraging item meta-data for sparsity reduction. In *AAAI international conference on artificial intelligence (AAAI)* (pp. 374–380).
- Ying, H., Chen, L., Xiong, Y., & Wu, J. (2016). Collaborative deep ranking: A hybrid pair-wise recommendation algorithm with implicit feedback. In *Pacific-Asia conference on knowledge discovery and data mining (PAKDD)* (pp. 555–567).
- Yu, Z., Xu, H., Yang, Z., & Guo, B. (2016). Personalized travel package with multi-point-of-interest recommendation based on crowdsourced user footprints. *IEEE Transactions on Human-Machine Systems*, 46(1), 151–158.
- Zheng, L., Noroozi, V., & Yu, P. S. (2017). Joint deep modeling of users and items using reviews for recommendation. In *ACM international conference on web search and data mining (ACM WSDM)* (pp. 425–434).
- Zhou, Q., Chan, C. W., & Tontiwachiwuthikul, P. (2009). A monitoring and diagnostic expert system for carbon dioxide capture. *Expert Systems with Applications*, 36(2), 1621–1631.