DynFluid: Predicting Time-Evolving Rating in Recommendation Systems via Fluid Dynamics

Huanyang Zheng and Jie Wu
Department of Computer and Information Sciences, Temple University, USA
Email: {huanyang.zheng, jiewu}@temple.edu

***Abstract*—In trust-based recommendation systems, if a user is predicted to have a high rating of a product, then this product is recommended to that user for shopping potential. Therefore, rating predictions are critical for qualified recommendations. In this paper, based on the fluid dynamics theory, we propose a novel rating prediction scheme called DynFluid. The key observation is that the rating of a user depends on his/her user experience, as well as the ratings of other users. For example, users may refer to friends' ratings upon rating a product, themselves. DynFluid analogizes the rating reference among the users to the fluid flow among containers: each user is represented by a container; the rating of a user is mapped to be the fluid temperature in the corresponding container. Two user characteristics, persistency and persuasiveness, are also incorporated into DynFluid. Finally, real data-driven experiments in Epinions and Ciao validate the efficiency and effectiveness of the proposed DynFluid.**

***Keywords*-Online social networks, recommendation systems, trust propagation, rating prediction, fluid dynamics.**

I. Introduction

Nowadays, increasing amounts of people are involved in Online Social Networks (OSNs) for completing daily activities, including forming an opinion on a particular product. One central issue in OSNs is the notion of *trust*. More specifically, trust in OSNs denotes the subjective probability by which a user expects another given user to perform a given action. One important application of trust is in online recommendation systems, such as Epinions and Ciao [1]. Such systems have two essential components: the rating (or opinion) of a user on a product and the trust relationships among users. The product rating of a user depends on his/her user experience, as well as the existing ratings of the other users (such as trusted friends).

In this paper, we focus on the rating prediction problem in trust-based recommendation systems [2]. If a user is predicted to have a high rating of a product, then the service provider can recommend this product to that user for shopping potential. As shown in Fig. 1, a typical online recommendation system provides the following product information to a user: a brief product description; a public broadcast channel (or simply *public channel*) that shows the average product rating of all users; the ratings and comments from the *trusted friends* of that user; the ratings and comments from strangers. The last one is not shown in Fig. 1, and is not considered in this paper. This is because very few users would read the ratings and comments from strangers. On the other hand, trusted friends' ratings and the public channel are likely to be referred to when users give out their own ratings.

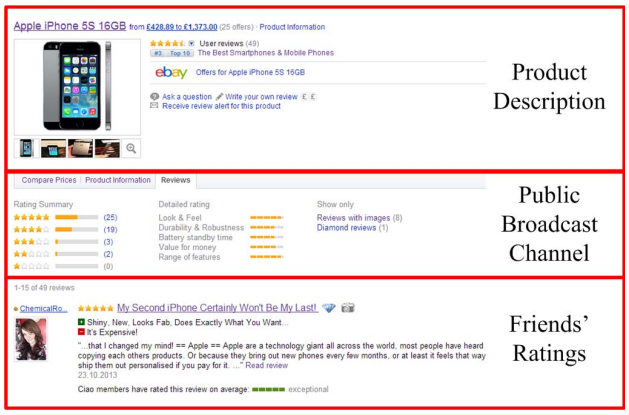


Fig. 1. An illustration for the rating of a product in Ciao.

Our rating predictions are based on the references of the ratings among the users. To further understand the rating behavior of a user, here we introduce two concepts called *persistency* and *persuasiveness*. Persistency denotes how much a user insists on his/her own user experience. A user with a low persistency would like to refer to the ratings of other users. On the other hand, persuasiveness denotes the convincingness of a user's rating. A user with a higher persuasiveness indicates that his/her ratings are more likely to be referred to by the other users. These two user characteristics are critical for improving the accuracy of the rating predictions.

We first consider the scenario with one product. A subset of users (raters, denoted as R) have prior ratings on the product. The remaining non-raters (denoted as N) have not rated the product, but they can refer to the existing ratings before giving out their own ratings. In other words, the non-raters are influenced by the raters, in terms of the ratings. An example for our system is shown in Fig. 2, where the numbers on the top of the raters are their prior ratings of the product. Directional links are trust relationships. Then, a directional link from a_1 to a_3 means that a_1 is trusted by a_3, and thus a_3 may refer to a_1's rating before giving out its own rating.

As shown in Fig. 2, our main idea is to analogize the rating reference (representing the opinion propagation) among the users to the fluid flow among the containers. The users are mapped to be containers, while the trust relationships among users are mapped to be directional pipes (pipes with one-way valves). The fluid temperature in a container represents the

978-1-4673-7952-6/15 $31.00 © 2015 IEEE
DOI 10.1109/Trustcom.2015.350

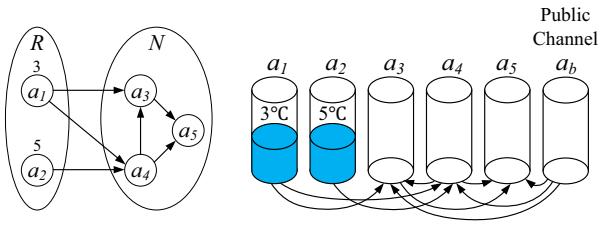


Fig. 2. An illustration of DynFluid. Each user corresponds to a container, while the directional pipes represent the trust relationships among users. The container a_b is added to represent the public channel.

rating of the corresponding user. For example, in Fig. 2, the rater a_1 has a rating of 3, and thus the corresponding fluid temperature is 3°C. Then, the persistency and persuasiveness of a user are represented by the fluid height and the cross-sectional area of the corresponding container, respectively. A higher fluid height indicates a larger persistency, and a larger cross-sectional area indicates a higher persuasiveness. This is because a container with a higher fluid height and a larger cross-sectional area can send out more fluid to change the ratings of its neighbors. Once there exists a fluid height difference between two connecting containers, fluid will gradually flow from one container to the other. These kinds of fluid dynamics represents how the users value their trusted friends' ratings in a time-evolving manner. To further incorporate the public channel, we add an extra container (the container a_b in Fig. 2) into the system. This extra container connects to all the non-raters, since the public channel can be seen by all the users. We are very interested in the impact of the public channel, compared to the impact of the trusted friends. For one user, is the public channel more trustworthy than one of the trusted friends? This question will be explored in our paper.

Our results and contributions are summarized as follows:

- We propose a clean-slate rating prediction method, DynFluid, to capture the time-evolving ratings. DynFluid considers both the influence from the public channel and the ratings from trusted friends.
- We introduce the two concepts of persistency and persuasiveness, which reveal the users' rating behaviors. The fluid dynamics theory is used to model the rating process.
- Real data-driven experiments in Epinions and Ciao are conducted to evaluate the proposed DynFluid. The results are shown from different perspectives to validate the efficiency and effectiveness of our approach.

The remainder of this paper is organized as follows: Section II surveys related work. Section III states the model and then formulates the problem. Section IV describes DynFluid, including its analogy insights and algorithmic properties. Section V includes extensive real data-driven experiments. Finally, Section VI concludes this paper.

II. Related Work

In this section, we review the literature of trust models, trust propagations, and ratings in the recommendation systems.

Trust models. Currently, a wide range of disciplines have examined various issues related to trust [3]. However, there is no consensus on how trust should be defined. In [4–6], trust in a person is defined as a commitment to an action, based on a belief that the future actions of that person will lead to a good outcome. Here, trust is subjective and personalized. We consider trust to be asymmetric. A user may trust another user more than he is trusted back. Another closely-related concept is reputation, which is usually an objective measure. One may trust a stranger if he/she has a strong reputation [7].

Trust Propagations. Generally speaking, the trust follows the principle of transitivity [3]. Trusts among users form a trusted graph. Sun et al. [8] proposed an information-theoretic framework on trust propagation by stating two axioms as possible guiding principals: (1) concatenation propagation of trust does not increase trust, and (2) multipath propagation of trust does not reduce trust. The existing path-based propagation methods include the Dempster-Shafer combination rule [9], serial-parallel merge [10] using subjective logic, and path concatenation [11] from path algebra. Several models have been proposed for graph-based propagation. Both MoleTrust [5] and TidalTrust [4] are based on breadth-first search. TidelTrust selects the strongest shortest path, while MoleTrust uses the hop count (also called horizon) to control the length of the selected path. However, these approaches are information-lossy, while some more interesting approaches are graph analogy-based. In [12], a generalized reliability theory is applied to a trusted network with failure-prone elements. RelTrust [13] emulates a trusted graph with a resistive network, using a logarithmic function to map the trust values to the resistance values.

Ratings. In rating-based systems, the rating (or opinion) of a user is usually a numeric value on an online website [1]. Anderson et al. [14] introduced a finite integer set with $\{+, -, 0\}$ representing positive, negative, and neutral ratings. The predictions of positive ratings are more useful than those of negative ratings. This is because only products with predictions of positive ratings are recommended to the users. In our model, the rating is measured by the fluid temperature, which can be easily updated based on the fluid dynamics theory. Zhu et al. [15] found that a person's opinion is significantly swayed by others' opinions. Our DynTrust takes [16] as a foundation: when new opinions come, each person refines his/her opinion through exchanging opinions with friends.

III. Model and Problem Formulation

In this paper, we focus on the rating prediction problem in trust-based recommendation systems. Accurate predictions can help the service provider recommend appropriate products to the user for shopping potential. Generally speaking, the rating of a user depends on his/her user experience, as well as the ratings of other users. For example, users may refer to the ratings of trusted friends upon their own ratings (i.e., the opinion propagations among users). However, user experience depends on many external unknown factors, and thus is hard to predict. Therefore, we predict the rating of a user, based on the ratings of his/her trusted friends and the public channel.

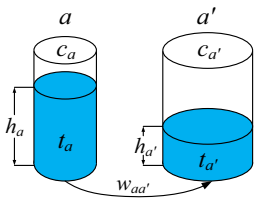


Fig. 3. A motivational example to illustrate the analogy insights.

To model the rating behavior of a user, the concepts of persistency and persuasiveness are introduced into our scheme. Persistency denotes how much a user relies on his/her own user experience. A user with a low persistency would like to refer to the ratings given out by the other users (including the trusted friends and the public channel). On the other hand, persuasiveness denotes the convincingness of a user's rating. A user with a higher persuasiveness means that his/her ratings are more likely to be referred to by the other users. Our scenario is based on a directed graph $G = (V, E)$, where V is a set of nodes (i.e., users), and $E \subseteq V^2$ is a set of directed edges (i.e., trust relationships). The edge $e_{aa'}$ has the direction from the node a to a', indicating that a is trusted by a', and thus a' may refer to a's rating before giving out his/her own rating. The node set V can be divided into two subsets: a subset of raters, R, and a subset of non-raters, N. The raters have given out their ratings on the product, and they cannot change their ratings anymore. The non-raters have not rated the product, but they may refer to the existing ratings to give out their own ratings, based on the persistency and persuasiveness of the users. The objective of this paper is to predict the ratings of non-raters in a time-evolving manner.

IV. DynFluid: Algorithm Details

A. Analogy Insights

Our main idea is that the rating reference process (or opinion propagation process) is analogous to the fluid flow. The users are modeled as containers, while the trust relationships among users are modeled as directional pipes (i.e., pipes with one-way valves). All the containers are placed on the same horizontal plane. The containers are large enough to hold the fluid (the fluid will not overflow). Pipes are installed at the bottom of the containers, and may have different pipe sizes (the cross-sectional areas of pipes). For a pipe from a to a', its pipe size is denoted as $w_{aa'}$, which represents the strength of the corresponding trust relationship. A larger pipe size indicates a higher level of trust. As shown in Fig. 2, the fluid temperature in a container represents the rating of the corresponding user. Then, the persistency and persuasiveness of a user are represented by the fluid height and the cross-sectional area of the corresponding container, respectively. For the user a, the corresponding fluid temperature, fluid height, and the cross-sectional area is denoted by t_a, h_a, and c_a, respectively. A higher fluid height indicates a larger persistency, while a larger cross-sectional area indicates a higher persuasiveness.

To capture the analogy insights, a motivational example is provided in Fig. 3, where we have two containers (users) of a and a'. Then, we have four analogy insights. (1) The height of the fluid in a' is lower than that in a. Therefore, the fluid in a flows into a' through the one-way pipe, meaning that a' refers to the rating given out by its trusted friend a. In other words, a' does not insist on its own user experience (i.e., a low persistency), and thus it wants to refer to the ratings of its trusted friend for its own rating. (2) Since a has a very small cross-sectional area (i.e., a low persuasiveness), only a small volume of fluid would eventually flow from a to a' at the end. This means that the rating of a is not very convincing, and thus, it only slightly changes the rating of a'. (3) The fluid from a mixes up with the existing fluid in a', representing that a' refines its own opinion in a time-evolving manner. The fluid temperature (the rating) of a' is changed by the fluid from a. (4) The pipe size has some impacts on the duration of the fluid flowing time. If $w_{aa'}$ is large, then the fluid in a flows into a' within a short time. This means that a' updates its rating more quickly, if a' trusts a more. These four analogy insights show that the fluid flow can capture the rating reference process (or opinion propagation process) in a decent way.

B. Fluid Update Principles

Discrete Approach. In the real world, the fluid flow is a continuous-time system, which is hard to compute. Hence, the discrete approach is used. The continuous physical time is discretized into a series of time slots. The duration of each time slot is denoted as Δt. Fluid flows are described as a set of partial differential equations. We consider that the fluid update is performed synchronously at the end of each time slot. The update process is shown in Fig. 4, which corresponds to the example in Fig. 2. Let R denote the total number of fluid updates. At the beginning of the i^{th} time slot, we prepare the fluid update and check whether the fluid will flow in each pipe, by comparing the fluid heights of two connected containers. If there is a directional pipe from a to a', and the fluid height in a is higher than that of a', the fluid will flow from a to a'; if either of the two conditions do not meet, no fluid will flow. The first condition means that a is trusted by a', and thus, the rating of a may be referred to by a'. The second condition means that a' has a low persistency, and thus a' wants to refer to the rating of a. We record the volume and temperature of the flowing fluid. At the end of each time slot, we mix up the flowing fluid and the remaining fluid as the fluid updates.

Initialization. The fluid heights of all the non-raters are initialized to be zero, since they have not rated the product yet. The fluid height of each rater is initialized according to the persistency of that rater. The fluid temperatures of all the raters are initialized according to their ratings, as shown in Fig. 2. The cross-sectional area of each user is initialized based on the persuasiveness of that user. As for the public channel, it is modeled as an extra container that connects to all the non-raters. The initialization of the public channel is similar to that of the raters, the only exception being that its fluid temperature is set as the average fluid temperature among all the raters.

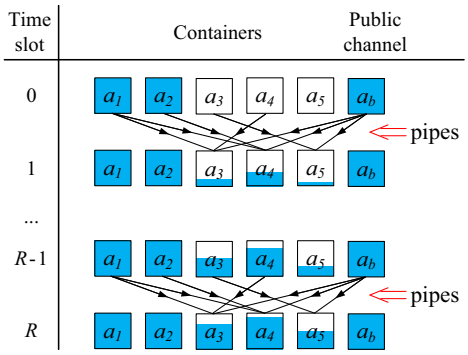


Fig. 4. A discrete approach to compute the fluid flow.

Update Details. First, let us consider a single pipe, say the pipe connecting a and a', with cross-sectional area $w_{aa'}$. This scenario has been shown in Fig. 3. At the beginning of the i^{th} time slot, if a has more fluid than a', fluid will flow from a to a' during this time slot of duration Δt. Based on the Bernoulli's principle [17], the speed of fluid flowing at the bottom of a will be $2g\sqrt{h_a(i) - h_{a'}(i)}$. Here, g is the gravitational acceleration, which is a constant. Hence, the volume of fluid that flows from a to a' in the i^{th} time slot is:

$$s_{aa'}(i) = 2g\sqrt{h_a(i) - h_{a'}(i)} \times w_{aa'} \times \Delta t \tag{1}$$

The insight behind Eq. 1 is that, if a' has a lower persistency and a is trusted more by a', then the rating of a is more valuable to a'. In addition, the fluid temperature that corresponds to $s_{aa'}(i)$ is denoted as $t_{aa'}(i)$, which is equal to the temperature of the fluid in the container a. Let $s_a(i)$ denote the volume of the fluid in a at the i^{th} time slot. Then, for the $(i+1)^{th}$ time slot, the volume of the fluid in a can be calculated as:

$$s_a(i+1) = s_a(i) - \sum_{a' \in N_a^+} s_{aa'}(i) + \sum_{a' \in N_a^-} s_{a'a}(i) \tag{2}$$

where N_a^+ and N_a^- are the outgoing and incoming neighbors of a, respectively. The fluid height in a can be calculated by $h_a(i+1) = s_a(i+1)/c_a$, where c_a is the cross-sectional area of the container a. The cross-sectional area of a represents its persuasiveness, which has an impact on the final ratings of its outgoing neighbors. This is because a larger cross-sectional area will lead to a smaller fluid height change, after the current update. If a has a larger cross-sectional area, more fluid will eventually flow into its outgoing neighbors.

Let $t_a(i)$ denote the temperature of the fluid in a at the i^{th} time slot. According to the fluid mixing formula [17], the fluid temperature after being mixed up is:

$$t_a(i+1) = \frac{[s_a(i) - \sum_{a' \in N_a^+} s_{aa'}(i)] \cdot t_a(i) + \sum_{a' \in N_a^-} [s_{a'a}(i) \cdot t_{a'a}(i)]}{s_a(i+1)} \tag{3}$$

Eq. 3 is essentially $\sum(\text{volume} \cdot \text{temperature}) / \sum \text{volume}$. The first part of the numerator represents the remaining fluid in container a, while the second part corresponds to the incoming fluid flowing from the other containers. The denominator is the volume of the mixed fluids, as shown in Eq. 2.

Algorithm 1 DynFluid

Input: The directed graph G and the existing ratings; The parameters for the initialization; Δt, R, and g for the fluid flow update;
Output: The predicted ratings of the non-raters;

1: Associate each node in G with a container;
2: Associate each edge in G with a directional pipe;
3: Add an extra container (i.e., public channel) that connects to all the non-raters' containers.
4: Initialize the fluid height and temperature in each container;
5: **for** $i = 0$ to $R - 1$ **do**
6: **for** each pipe from a to a' **do**
7: **if** $h_a(i) > h_{a'}(i)$ **then**
8: Calculate the volume and temperature of the outgoing flowing fluid, based on Eq. 1;
9: **for** each rater's container and the extra container **do**
10: Inject additional fluid to maintain the fluid height and temperature;
11: **for** each non-rater's container **do**
12: Mix up the incoming flowing fluid, as to update the fluid height and temperature, based on Eqs. 2 and 3;
13: **return** the fluid temperature in the non-rater's container.

Considering that the raters have given out their ratings (i.e., they no longer update their ratings), the fluid heights of the raters are fixed. This can be viewed as giving additional fluid injections to the raters. The raters will no longer refer to the ratings of the other users, and their ratings are only referred to by the non-raters. As for the public channel, its fluid height is also fixed. If a user has given out his/her rating, then this user can no longer change his/her rating on the public channel.

C. Algorithm Overview and Time Complexity Analysis

DynFluid is shown in Algorithm 1. Lines 1 and 2 associate nodes and edges in G with containers and pipes, respectively. In line 3, an extra container is added to represent the public channel. Line 4 shows the initialization process. Lines 5 to 12 show the discrete approach for calculating the fluid flow. The continuous physical time is discretized into R time slots, i.e., R rounds of fluid updates. In each round, we first calculate the volume of the flowing fluid in each pipe, as shown in lines 6 to 8. In lines 9 and 10, we inject additional fluid to the containers that correspond to the raters and the public channel. At the end of each round, we update the fluid in the non-raters' containers (lines 11 and 12). Finally, in line 13, the fluid temperature in the non-rater's container is returned as the predicted rating.

The intialization of Algorithm 1 (lines 1 to 4) takes a time complexity of $O(V + E)$. Each round of fluid update in lines 5 to 12 also takes a time complexity of $O(V + E)$. Since fluid updates have R rounds, the total time complexity of Algorithm 1 is $O(R \cdot (V + E))$. Considering that social networks are generally sparse and R should be a small constant, the proposed DynFluid could be very efficient for real-world rating predictions that involve millions of users.

D. Convergence Analysis

In this subsection, we focus on the convergence of the proposed DynFluid. First, we have the following theorem:

***Theorem** 1:* If we use a constant value (denoted by h) to initialize the fluid heights of all the raters and the public channel, then the fluid heights of all the non-raters will always be no larger than h, during the fluid updating process.

Proof: We proof this theorem by contradiction. Suppose there is a non-rater a, whose fluid height h_a is larger than h. There are two possible cases for explaining a's fluid height. The first case is that the fluid in a comes directly from a rater (or the public channel) a', where $h_{a'} > h_a > h$. However, the fluid heights of all the raters (or the public channel) are h, since they only give out their fluids to non-raters. Therefore, the first case contradicts the assumption, which should be invalid. The second case is that the fluid in a comes directly from a non-rater. However, working iteratively with this case will eventually result in a non-rater whose fluid comes directly from a rater. Therefore, the second case will be reduced to the first case, which is not valid by contradiction. ■

***Theorem** 2:* If we use a constant value (denoted by h) to initialize the fluid heights of all the raters and the public channel, then, after a time period that is sufficiently long, the fluid heights of all the non-raters will be h.

Proof: Suppose there is a non-rater a. According to Theorem 1, we have $h_a \leq h$. If $h_a = h$ for any non-rater a, then the proof completes. If $h_a < h$, we also have two cases. The first case is that the fluid in a comes directly from a rater (or the public channel) a'. This means that $h_{a'} = h > h_a$. Since there is a pipe from a' to a, the flowing fluid in this pipe will eventually fill up the height gap between a' and a, meaning that we have $h_a = h$ after a sufficiently long time period. The second case is that the fluid in a comes directly from a non-rater. Iteratively doing this case will eventually reduce this case to the first case, since the fluids of all the non-raters originate from the raters and the public channel. ■

Theorems 1 and 2 show that the DynFluid will converge after certain rounds of fluid updates. It can be explained by the rating reference process in the real world. Initially, a user has no idea about the given product. Upon referring to the ratings of the other users, this user formulates and refines his/her own rating in a time-evolving manner. During this process, a person's opinion becomes more and more mature, indicating increased persistency. Therefore, this phenomenon is consistent with our real-world experiences.

E. Algorithm Properties

In this subsection, we study the properties of the proposed DynFluid. First, we have the following property:

***Property** 1:* In DynFluid, the opinion influence from a user, a, to another user, a', decays monotonously with respect to the hop-count distance from a to a'.

This property indicates that a user is influenced more by his/her trusted friends and the public channel than strangers. In DynFluid, the 1-hop neighbor of a non-rater can pass their fluids directly to this non-rater, during the 1^{st} round of fluid updates. Meanwhile, the k-hop neighbor can only pass his/her fluids to that non-rater during the k^{th} round of fluid updates. The fluids from nearby neighbors arrive at the non-rater earlier (in terms of the discrete time slot) with a larger volume, since the fluid height of the non-rater is lower at an earlier time. The fluid from strangers arrives at that non-rater later with a smaller volume, since the fluid height of the non-rater becomes higher at a later time. The ratings of trusted friends and the public channel are more valuable than the strangers' ratings. Then, another property of DynFluid is:

***Property** 2:* In DynFluid, the certainty of the rating prediction for a non-rater can be measured by the fluid height (or persistency) of that non-rater.

This property states that the certainty of the rating prediction is highly related to the persistency of the corresponding non-rater. DynFluid shows how a user refines its rating in a time-evolving manner. At the beginning, the user has a low persistency, and thus he/she receives multiple opinions from his/her trusted friends and the public channel. As time goes by, the opinion of a user becomes more and more mature, indicating that the persistency of that user gets higher and higher. Therefore, the rating of a user with a higher persistency is more stable than the one with a lower persistency, representing that the certainty can be measured by the persistency.

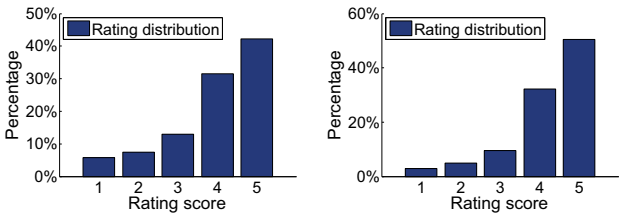
V. Evaluations

A. Basic Settings

Dataset information. In our experiments, the datasets of Epinions and Ciao [18] are used. These two datasets are collected online from www.epinions.com and www.ciao.com. These two datasets include the directional trust relationships among users, as well as the users' ratings (recorded as rating scores from 1 to 5) on some products. The distributions of rating scores in Epinions and Ciao are shown in Fig. 5. It can be seen that users are more likely to give out high ratings. More than 40% of rating scores are 5 (the highest rating score). Then, the Epinions dataset consists of 49,290 users who rated a total of 139,738 different products. The total number of issued trust relationships is 487,181. The Ciao dataset consists of 2,248 users who rated a total of 16,861 different products. The total number of issued trust relationships is 57,544.

Since the ratings on a specified product are generally sparse with respect to the graph size, a user may not have a trusted friend who has rated the same product as he/she did. Therefore, we do not run experiments directly on the whole dataset. Alternatively, given a product, we extract subgraphs to test rating predictions: if a user does not rate that product, or this user does not have a trusted friend who has rated that product, then this user is not considered in our experiments; otherwise, we generate a subgraph centered on that user to predict his/her rating. This subgraph is composed of all neighbors of that user within a certain hop count. We use the 1-hop, 2-hop, and 3-hop subgraphs. A larger subgraph provides more information, and thus a better performance should be obtained by the DynFluid.

Performance metric. To measure the prediction errors of the proposed DynFluid, the leave-one-out method [19] is used.



(a) Rating scores in Epinions. (b) Rating scores in Ciao.

Fig. 5. The distribution of the rating scores.

For the generated subgraph of a specific user, the groundtruth rating of that user is masked and then predicted based on the generated subgraph. We compare the predicted rating with the masked groundtruth rating, the difference between which is the prediction error. To measure the prediction errors among different subgraphs and different products, we adopt the root mean squared error (RMSE) [6], which is the root of the mean squared prediction error among all the generated subgraphs and all the products. RMSE represents the standard deviation of the differences between predicted ratings and groundtruth ratings. A smaller RMSE indicates a better prediction.

The second performance metric is the classic F-score, which is the harmonic mean of precision and recall:

$$\text{F-score} = \frac{2TP}{2TP + FP + FN} \tag{4}$$

When the groundtruth is that the corresponding user has a rating score of no less than three (called positive rating), TP and FP (true and false positive cases) are the numbers of correct and incorrect predictions, respectively. Then, the false negative case, FN, is the number of incorrect predictions, when the groundtruth is that the corresponding user has a rating score of less than three (called negative rating). Note that a larger F-score indicates a better prediction.

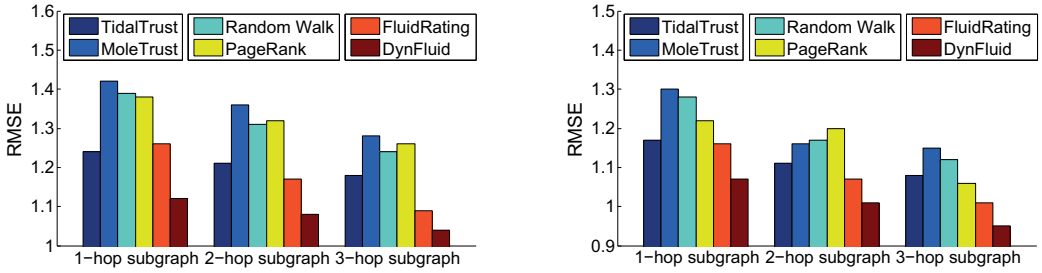
Default parameters. In our experiments, the subgraph for rating predictions is generated, in default, by all neighbors within 3 hops of a given user (also called 3-hop subgraph). Unless specified, we use $\Delta t = 0.1$ as the duration of each time slot and $R = 10$ as the rounds of fluid updates. The fluid heights of all the raters and the public channel are 10. The cross-sectional areas of all the containers are 1. The fluid temperature of each rater is initialized to be its rating score.

B. Comparisons with the Other Methods

In this subsection, we compare DynFluid with the other state-of-the-art algorithms in terms of RMSE and F-score. Comparison algorithms include TidalTrust [4], MoleTrust [5], Random Walk [6], PageRank [14], and FluidRating [16]. (1) TidalTrust finds all trusted raters through the shortest path to the given user, and then aggregates their ratings as the predicted rating of that given user. (2) MoleTrust has two steps [20]. In the first step, it takes two given nodes (source and destination) as the input and then builds a directed trust graph from the source to the destination. In the second step, it walks through the directed trust graph and calculates the trust values of the visited nodes. The rating of the destination node serves as the predicted rating of the source node. (3) Random Walk aggregates the ratings of users arriving by random walks from a given user as the predicted rating of that given user [6]. (4) PageRank computes the user's reputation for trust propagations. We take its result when it converges. (5) FluidRating is the foundation of this work. However, FluidRating does not consider the public channel, which is really impactful for the rating predictions.

The comparison results are shown in Figs. 6 and 7. The former one shows the results with the RMSE metric and the later one shows the results with the F-score metric. A smaller RMSE means a better result, while a larger F-score means a better result. Figs. 6(a) and 6(b) show the results for Epinions and Ciao, respectively. The left part of Fig. 6(a) is the result for the subgraphs that are generated by all neighbors within 1 hop of the given user, while the middle and right parts are those within 2 and 3 hops, respectively. DynFluid outperforms all the other algorithms, since it considers the ratings of the trusted friends and the public channel. For a 1-hop subgraph, DynFluid has a more than 10% improvement with respect to all the other methods. For a 3-hop subgraph, DynFluid has a more than 5% improvement over FluidRaing, and an over 10% improvement compared to the other methods. DynFluid performs better for subgraphs of neighbors within 3 hops than those within 1 hop. When the subgraph is larger, users can refer to more rating scores from their trusted friends and make a better rating. The performance gap between DynFluid and FluidRating gets smaller, when the generated subgraph is larger. This is because a larger subgraph brings more information on the public channel. The results in Ciao are similar to those in Epinions. However, the overall RMSE in Ciao is lower than that in Epinions (about 10% lower).

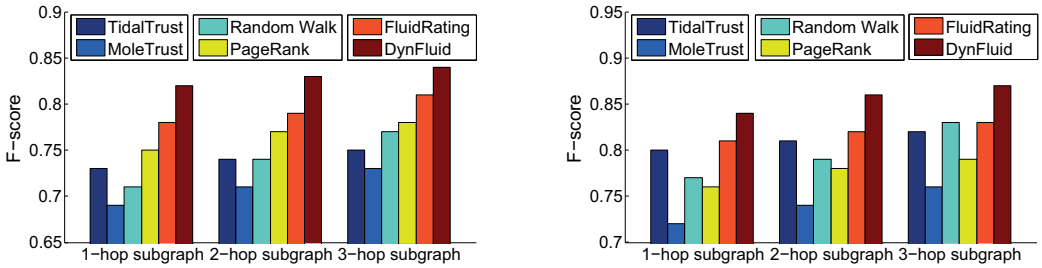
Fig. 7 shows the comparasion results with the F-score metric. Fig. 7(a) and 7(b) show the results for Epinions and Ciao, respectively. In Fig. 7(a), DynFluid has a significant performance improvement, compared to Random Walk (about a 15% higher F-score). For the other methods, a performance improvement that is larger than 5% is also obtained by DynFluid. According to the definition of F-score, DynFluid can accurately predict the true positive case, where the groundtruth is that the corresponding user has a rating score of no less than three (positive rating). Note that the true positive case is the most important case for DynFluid, since its motivation is the trust-based recommendation. If a user is predicted to have a high rating on a specified product, then the service provider can recommend this product to that user for the shopping potential. Although F-score does not consider the true negative case, this case is not important for recommendations. This is because the service provider should not recommend a product to a user, if this user is predicted to have a negative rating on that product. DynFluid has an F-score of about 85% when we use the 3-hop subgraph. Therefore, DynFluid is qualified for trust-based recommendations. The F-score of DynFluid in Ciao is a little bit higher than that in Epinions, indicating that the rating scores in Ciao are more predictable in the F-score metric. This is consistent with that in the RMSE metric.



(a) The RMSE in the Epinions dataset.

(b) The RMSE in the Ciao dataset.

Fig. 6. Compare DynFluid with the other methods, in terms of the RMSE metric.



(a) The F-score in the Epinions dataset.

(b) The F-score in the Ciao dataset.

Fig. 7. Compare DynFluid with the other methods, in terms of the F-score metric.

C. The Impact of The Public Channel

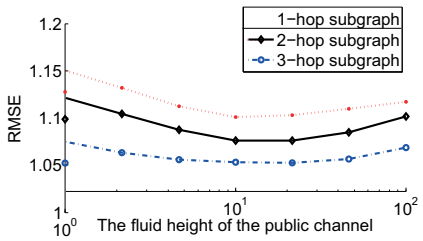
In the previous subsection, we compare DynFluid with other state-of-the-art methods. Here, we conduct additional experiments to further understand the impact of the public channel. Instead of the default setting, where the fluid height of the public channel is initialized to be 10, we tune the fluid height of the public channel, as to observe the RMSE variance of the DynFluid. Note that the fluid heights of all the raters are set to be 10. A higher fluid height of the public channel means that it has a larger impact on the users.

The results are shown in Fig. 8, where we conduct the above experiments for DynFluid in subgraphs generated by all neighbors within 1 hop, 2 hops, and 3 hops, respectively. Fig. 8(a) and 8(b) show the results for Epinions and Ciao, respectively. In both datasets, the RMSE of DynFluid first decreases and then increases, with respect to the initial fluid height of the public channel. When the initial fluid height of the public channel is 1, the DynFluid mainly uses the ratings of the trusted friends for rating predictions, leading to a high RMSE. In other words, if the impact of the public channel is ignored, then the rating prediction becomes inaccurate. On the other hand, when the initial fluid height of the public channel is 100, the DynFluid mainly uses the public channel for rating predictions, which also leads to a high RMSE. This means that the ratings of the trusted friends are also not negligible. Note that the RMSE of the DynFluid goes to the minimum, when the initial height of the public channel is set to be about 10 to 20. Meanwhile, the initial heights of all the raters are 10. This observation implies that the public channel helps to improve the accuracy of the prediction. We also find that the impact of the public channel is more significant in 1-hop subgraphs than that in 3-hop subgraphs.

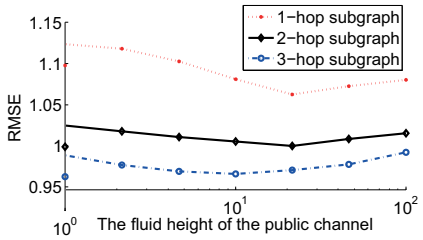
D. The Impact of The Persistency and Persuasiveness

In the previous experiments, we use a constant of 10 to initialize the fluid heights of all the raters. The cross-sectional areas of all the containers are set to be 1. Here, we initialize the fluid heights and the cross-sectional areas in a personalized way, as to see the impact of the persistency and persuasiveness. We estimate those two characteristics of a user through the total number of ratings (on different products) given out by that user. If a user has rated more products, we consider that user to have a higher persistency and persuasiveness, through a linear mapping process. This is because the user is more likely to insist on his/her own opinions, and is more likely to be an authority on the product for the other users.

First, we conduct experiments to observe the impact of the non-identical persistency, while the persuasiveness remains identical. The experimental results are shown in Fig. 9. It can be seen that the DynFluid with personalized persistency has a better performance than the DynFluid with the default setting (about 5% lower RMSE in both Epinions and Ciao). If we initialize the persistency to be identical, then a larger initial value or a smaller initial value has a very limited impact on the converged RMSE of the DynFluid. The variance caused by different initial values is less than 3%. This is because a higher fluid height also leads to a larger volume of flowing fluid as a self-regulation in the fluid flow system. DynFluid with personalized persistency has a slower convergence speed than that with identical persistency. This is because the user with a low persistency needs more time to refine his/her opinions. We also conduct experiments to observe the impact of the non-identical persuasiveness, while the persistency is identical. The experimental results are shown in Fig. 10. It can be seen that the DynFluid with personalized persuasiveness has a better

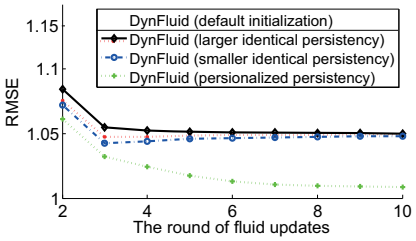


(a) The result for the Epinions dataset.

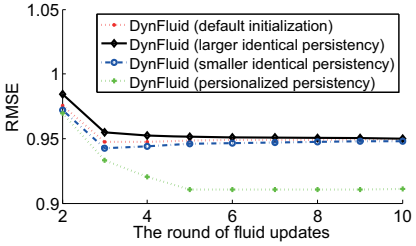


(b) The result for the Ciao dataset.

Fig. 8. The impact of the public channel.

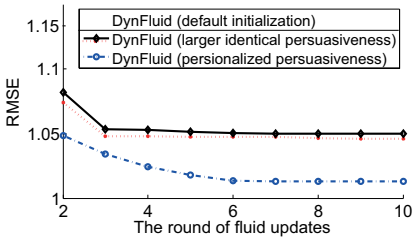


(a) The result for the Epinions dataset.

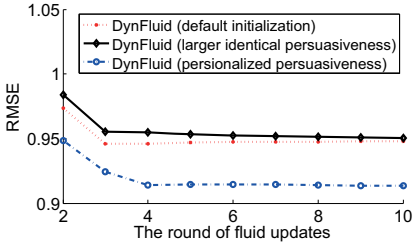


(b) The result for the Ciao dataset.

Fig. 9. The impact of the user persistency.



(a) The result for the Epinions dataset.



(b) The result for the Ciao dataset.

Fig. 10. The impact of the user persuasiveness.

performance (more than 5% lower RMSE) than the DynFluid with the default setting. If we initialize the persuasiveness to be identical, then its initial value has some impacts on the converged RMSE of the DynFluid. A too-large initial value leads to a performance degradation, since larger cross-sectional areas can weaken the impact of flowing fluid.

VI. Conclusions and Future Work

In this paper, we focus on the rating prediction problem in trust-based recommendation systems. The key observation is that the rating of a user depends on his/her user experience, as well as the ratings of other users. This paper proposes a novel rating prediction scheme called DynFluid, based on the fluid dynamics theory. DynFluid analogizes the rating reference among the users to the fluid flow among containers: each user is represented by a container; the rating of a user is mapped to be the fluid temperature. Two user characteristics, persistency and persuasiveness, are incorporated into DynFluid. Real data-driven experiments validate the performance of our DynFluid.

References

[1] J. Tang, H. Gao, and H. Liu, "mtrust: discerning multi-faceted trust in a connected world," in *Proceedings of ACM WSDM 2012*, pp. 93–102.

[2] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proceedings of ACM RecSys 2007*, pp. 17–24.

[3] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, 2007.

[4] J. A. Golbeck, "Computing and applying trust in web-based social networks," *PhD thesis, University of Maryland*, 2005.

[5] P. Massa and P. Avesani, "Trust metrics on controversial users: Balancing between tyranny of the majority," *IJSWIS*, vol. 3, no. 1, pp. 39–64, 2007.

[6] M. Jamali and M. Ester, "Trustwalker: a random walk model for combining trust-based and item-based recommendation," in *Proceedings of ACM SIGKDD 2009*, pp. 397–406.

[7] M. Sirivianos, K. Kim, and X. Yang, "Socialfilter: introducing social trust to collaborative spam mitigation," in *Proceedings of IEEE INFOCOM 2011*, pp. 2300–2308.

[8] Y. L. Sun, W. Yu, Z. Han, and K. R. Liu, "Information theoretic framework of trust modeling and evaluation for ad hoc networks," *IEEE JSAC*, vol. 24, no. 2, pp. 305–317, 2006.

[9] A. Jøsang and S. Pope, "Dempster's rule as seen by little colored balls," *Computer Intelligence*, vol. 28, no. 4, pp. 453–474, 2012.

[10] A. Jøsang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in *Proceedings of ACSC 2006*, pp. 85–94.

[11] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *Proceedings of ISWC 2003*, pp. 351–368.

[12] G. Mahoney, W. J. Myrvold, and G. C. Shoja, "Generic reliability trust model." in *Proceedings of PST 2005*, pp. 113–120.

[13] M. Taherian, M. Amini, and R. Jalili, "Trust inference in web-based social networks using resistive networks," in *Proceedings of IEEE ICIW 2008*, pp. 233–238.

[14] R. Andersen, C. Borgs, J. Chayes, U. Feige, A. Flaxman, A. Kalai, V. Mirrokni, and M. Tennenholtz, "Trust-based recommendation systems: an axiomatic approach," in *Proceedings of ACM WWW 2008*, pp. 199–208.

[15] H. Zhu, B. Huberman, and Y. Luon, "To switch or not to switch: Understanding social influence in online choices," in *Proceedings of ACM SIGCHI 2012*, pp. 2257–2266.

[16] W. Jiang, J. Wu, G. Wang, and H. Zheng, "Fluidrating: A time-evolving rating scheme in trust-based recommendation systems using fluid dynamics," in *Proceedings of IEEE INFOCOM 2014*, pp. 1707–1715.

[17] P. G. Hewitt, *Conceptual physics*. Pearson Educación, 2002.

[18] J. Tang, H. Gao, H. Liu, and A. D. Sarma, "eTrust: Understanding trust evolution in an online world," in *Proceedings of ACM SIGKDD 2012*, pp. 253–261.

[19] G. C. Cawley and N. L. Talbot, "Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers," *Pattern Recognition*, vol. 36, no. 11, pp. 2585–2592, 2003.

[20] P. S. Chakraborty and S. Karform, "Designing trust propagation algorithms based on simple multiplicative strategy for social networks," *Procedia Technology*, vol. 6, pp. 534–539, 2012.