Enhancing the Trust-Based Recommendation Process with Explicit Distrust

PATRICIA VICTOR and NELE VERBIEST, Ghent University
CHRIS CORNELIS, Ghent University and University of Granada
MARTINE DE COCK, Ghent University

When a Web application with a built-in recommender offers a social networking component which enables its users to form a trust network, it can generate more personalized recommendations by combining user ratings with information from the trust network. These are the so-called trust-enhanced recommendation systems. While research on the incorporation of trust for recommendations is thriving, the potential of explicitly stated distrust remains almost unexplored. In this article, we introduce a distrust-enhanced recommendation algorithm which has its roots in Golbeck's trust-based weighted mean. Through experiments on a set of reviews from Epinions.com, we show that our new algorithm outperforms its standard trust-only counterpart with respect to accuracy, thereby demonstrating the positive effect that explicit distrust can have on trust-based recommendations.

Categories and Subject Descriptors: H.3.3 [**Information Systems**]: Information Storage and Retrieval—*Information filtering*; H.4.2 [**Information Systems Applications**]: Types of Systems—*Decision support*

General Terms: Algorithms, Human Factors

Additional Key Words and Phrases: Distrust, recommender systems, social networks, trust metrics

ACM Reference Format:
Victor, P., Verbiest, N., Cornelis, C., and de Cock, M. 2013. Enhancing the trust-based recommendation process with explicit distrust. ACM Trans. Web 7, 2, Article 6 (May 2013), 19 pages.
DOI: http://dx.doi.org/10.1145/2460383.2460385

1. INTRODUCTION

The wealth of information available on the Web has made it increasingly difficult to find what one is really looking for. Although today it has become very easy to look up information, at the same time we experience more and more difficulties coping with this information overload. Hence, it comes as no surprise that personalization applications to guide the search process are gaining tremendous importance. One particular interesting set of applications that address this problem are online *recommender systems* [Adomavicius and Tuzhilin 2005; Burke 2002; Resnick and Varian 1997; Schafer et al. 1999; Uchyigit and Ma 2008]. Such systems use information about their users'

P. Victor thanks the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT) for funding her research. C. Cornelis thanks the Research Foundation - Flanders for funding his research.
Authors' addresses: P. Victor and N. Verbiest, Department of Applied Mathematics, Computer Science and Statistics, Ghent University, Krijgslaan 281 (S9), 9000 Gent, Belgium; C. Cornelis (corresponding author), Department of Applied Mathematics, Computer Science and Statistics, Ghent University, Krijgslaan 281 (S9), 9000 Gent, Belgium and Department of Computer Science and Artificial Intelligence, University of Granada, Calle del Periodista Daniel Saucedo Aranda s/n, 18071 Granada, Spain; email: chris.cornelis@ugent.be; M. de Cock, Department of Applied Mathematics, Computer Science and Statistics, Ghent University, Krijgslaan 281 (S9), 9000 Gent, Belgium.

DOI: http://dx.doi.org/10.1145/2460383.2460385

profiles and relationships to suggest items (books, movies, Web pages, travel packages, etc.) that might be of interest to them. Recommender systems can be used for several purposes, such as generating a ranking of items, recommending a sequence of items (think, e.g., of the personalized radio stations on Last.fm), or predicting the score of an item [Herlocker et al. 2004]. In this article, we focus on the latter type of systems, that is, recommenders that are used to accurately estimate the degree to which a particular user (the target user) will like a particular item (the target item). Note that such systems can also be of use for the ranking problem.

Most widely used recommendation systems are either content-based or collaborative filtering methods. Content-based systems tend to have their recommendation scope limited to the immediate neighborhood of a user's past purchase or rating record. For instance, if you have highly rated a romantic movie with Keanu Reeves, your next recommendation might be a romantic movie or a movie featuring Keanu. The system will continue to recommend related items only, and not explore your other interests. In this sense, recommender systems can be improved significantly by (additionally) using collaborative filtering [Resnick et al. 1994], which typically identifies users whose tastes are similar to yours and recommends items that these so-called "neighbors" have liked. This technique allows for more serendipitous recommendations: you might receive recommendations for movies in a genre that you are not familiar with but that are appreciated by your neighbors, so that there is a good chance that you will like them too. Collaborative filtering recommenders can be classified as either memory-based (heuristic-based) or model-based. The former generate recommendations that are based on the entire set of ratings that is available, while the latter only use the ratings to learn a model and then suggest items based on that model; think, for example, of clustering or matrix reconstruction techniques. For a discussion on model-based approaches, we refer to Adomavicius and Tuzhilin [2005]. The recommendation techniques that we will discuss in this article adhere to the memory-based, collaborative filtering paradigm.

Research has pointed out that people tend to rely more on recommendations from people they trust[1] than on online recommenders which generate recommendations based on anonymous people similar to them [Sinha and Swearingen 2001]. This observation, combined with the growing popularity of open social networks and the trend to integrate e-commerce applications with recommender systems, has generated a rising interest in *trust-enhanced recommendation systems*; see, for example, Golbeck and Hendler [2006], Hess and Schiedler [2008], Massa and Avesani [2007], O'Donovan and Smyth [2005], and Victor et al. [2009a]. These applications incorporate a trust network in which the users are connected by scores indicating how much they trust each other, and use that knowledge to generate recommendations: users can receive recommendations for items rated highly by people in their Web Of Trust (WOT), or even by people who are trusted by these WOT members, etc., yielding more, more accurate, and more personalized recommendations.

Many examples of trust-enhanced Web applications can be found; take for instance Moleskiing [Massa et al. 2005], a ski mountaineering community site which uses Friend Of A Friend-files[2] that contain trust information on a scale from 1 to 9 [Golbeck et al. 2003], or the e-commerce site Epinions.com which ranks reviews based on a trust

[1]As trust is used in a wide range of application domains, plenty of trust definitions exist. Many of them focus on a different aspect of trust, or stem from a different background (e.g., social sciences versus agent theory). In this article, we adopt the general definition of Jøsang and Lo Presti [2004], who consider it as the extent to which one is willing to depend on somebody in a given situation; in our case, we focus on trust that is explicitly given as a numerical score by the users of a recommender system.

[2]FOAF-files are machine-readable documents describing basic properties of a person, including links between the person and objects/people they interact with.

network that it maintains by asking its users to indicate which members they trust (i.e., their personal web of trust). Another well-known example is Golbeck's FilmTrust [Golbeck 2006], an online social network combined with a movie rating and review system in which users are asked to evaluate their acquaintances' movie tastes on a scale from 1 to 10.

One of the main strengths of this group of systems is their use of *trust metrics*, a set of mechanisms to estimate the trust between two unknown users in the network. The two key building blocks of any trust metric are trust propagation and aggregation operators: if a user a wants to form a trust opinion about an unknown user x, a has to inquire about x with one of his own trust relations, say b, who in turn might consult a trust connection, etc., until a user connected to x is reached. The process of predicting the trust score along the thus constructed path from a to x is called *trust propagation*. Since it often happens that a has not one, but several trust connections that it can consult for an opinion on x, we also require a mechanism for combining several trust scores originating from different sources; this process is called *trust aggregation*. By using trust propagation and aggregation operators, the sparsity problem of classic collaborative filtering algorithms can partially be solved since the trust metrics enable us to match the target user with a larger number of users who have rated the target item. Furthermore, research has also shown that trust-enhanced recommenders using trust propagation can alleviate the user cold start problem [Massa and Avesani 2004].

In this work we will focus on trust-aware approaches that mine a trust network consisting of explicitly issued trust statements, which gives us the opportunity to benefit from trust propagation and aggregation. The references given before clearly illustrate the popularity and increasing importance of trust-enhanced recommendation research. However, apart from trust, in a large group of users, each with their own intentions, tastes, and opinions, it is only natural that also *distrust* begins to emerge. For example, Epinions first provided the possiblity to include users in a personal WOT (based on their quality as a reviewer), but later on also introduced the concept of a personal "block list", which reflects the members that are distrusted by a particular user. The information in the WOT and block list is then used to make the ordered list of presented reviews more personalized. Another example of a Web application that also works with negative evaluation concepts is the technology news Web site Slashdot[3], which lets its users tag each other as "friends", "fans", "foes" or "freaks".

The more platforms and possibilities enabling users to express distrust, and the more users issuing distrust statements, the more important it will become to also tap this new information source. However, how to do this is still an open question. So far, only very few attempts have been made to actively incorporate distrust in the trust modeling as well as the recommendation domain (see, e.g., Guha et al. [2004], Jøsang [2001], Ma et al. [2009], and Ziegler and Lausen [2005]); this is the case for theoretical studies, but even more so for practical evaluations. This is due to several reasons, the most important ones being that very few datasets containing ratings, trust, and distrust information at once are available, and that there is no general consensus yet about how to propagate it and to use it for recommendation purposes. Ma et al. [2009] were the first to demonstrate that the incorporation of distrust information can be beneficial to recommendations, using a model-based approach. In this work, we want to evaluate whether this observation also holds in memory-based approaches, which make up the mainstream of trust-based methods currently available in the literature. In Victor et al. [2011d], we reflected a first time upon the effect of incorporation of distrust in recommendation formulas, but only mixed results were obtained. Therefore, the main

[3]See slashdot.org and www.essembly.com.

goal of this article is to further explore the potential of distrust for trust-enhanced recommenders, to experimentally evaluate its possible benefits, and as such to come to a better understanding of the role that distrust can play in future social network applications, and recommender systems in particular.

Since trust metrics are one of the main ingredients of any good trust- and distrust-enhanced recommendation algorithm, in Section 2 we first recall preliminaries about the trust and distrust propagation and aggregation techniques that we will use in the remainder of the article. In Section 3, we then embark upon the problem of distrust-aware recommendations. We start by discussing possible roles of distrust, in particular as a debugger of a web of trust. Subsequently, we introduce a novel distrust-enhanced algorithm and discuss related approaches. In Section 4, we investigate its performance experimentally; due to the inexistence of real-life datasets containing gradual trust and distrust values, our analysis will explore only the bivalent case, that is, in which only trust and distrust values of 0 and 1 occur. In particular, we shall work with a dataset from Epinions containing controversial reviews. These are reviews that receive a variety of high and low ratings, and hence are very challenging to accurately recommend. We illustrate that the choice of debug method can have a major impact on the generated recommendations, and show that the distrust-enhanced algorithm can produce more accurate recommendations than its trust-only counterpart (Golbeck's trust-based weighted mean) without a significant coverage loss. These results clearly demonstrate that the incorporation of distrust can indeed enhance the trust-based recommendation process.

2. TRUST METRICS

Before one can start computing with trust values and reasoning about propagation and aggregation, one first needs to agree on a trust model, so that the interpretation of a trust value is fixed. In Victor et al. [2009a] we motivated our choice to represent trust and distrust as two distinct but dependent gradual concepts that are not opposites of each other. In particular, we argued that trust networks are typically challenged by two important problems influencing trust opinions. First, in large networks it is likely that many users do not know each other, hence there is an abundance of ignorance. Second, because of the lack of a central authority, different users might provide different and even contradictory information, hence inconsistency may occur. Models working with only a linear scale of trust values cannot cope with this kind of situation. For instance, a trust value of 0 might be interpreted as an indication of either active distrust in another user, or simply of ignorance about this user. For this reason, we advocated a trust model in which trust scores are (trust,distrust)-couples drawn from a bilattice [Ginsberg 1988].

We model a trust network as a directed graph with the users as nodes, and directed trust links as edges.

Definition 2.1 (*Trust Network, Trust Score*). [Victor et al. 2009a]. A trust network is a couple (A, R) in which A is the set of users and R is an $A \times A \rightarrow [0, 1]^2$ mapping that associates with each couple (x, y) of users in A a trust score $R(x, y) = (t, d)$ in $[0, 1]^2$, in which t is called the trust degree and d the distrust degree.

In other words, a trust score represents both the trust and distrust relation between two agents. Trust scores in our setting are interpreted as epistemic values: the trust and distrust degrees are not complementary, but they reflect the imperfect knowledge we have about the actual trust and distrust values (which are complementary).

Bilattice theory enables us to compare the trust scores in several ways.

Definition 2.2 (*Trust-Distrust Ordering, Knowledge Ordering*). [Victor et al. 2009a]. The trust-distrust $\leq_{td}$ and knowledge ordering $\leq_k$ are defined by

$$(t_1, d_1) \leq_{td} (t_2, d_2) \text{ iff } t_1 \leq t_2 \text{ and } d_1 \geq d_2$$
$$(t_1, d_1) \leq_k (t_2, d_2) \text{ iff } t_1 \leq t_2 \text{ and } d_1 \leq d_2$$

for all (t_1, d_1) and (t_2, d_2) in $[0, 1]^2$.

The lattice $([0, 1]^2, \leq_{td})$ orders the trust scores going from complete distrust $(0, 1)$ to complete trust $(1, 0)$. The lattice $([0, 1]^2, \leq_k)$ evaluates the amount of available trust evidence, ranging from a "shortage of evidence", namely, $t_1 + d_1 < 1$, to an "excess of evidence", namely $t_1 + d_1 > 1$; the value $t_1 + d_1$ is also called the knowledge degree of the trust score (t_1, d_1). The boundary values of the $\leq_k$ ordering, $(0, 0)$ and $(1, 1)$, reflect ignorance, respectively, contradiction.

We will use trust scores to compare the degree of trust and distrust a user may have in other users in the network, or to compare the uncertainty that is contained in the trust scores. This information can for example, be used in the ranking mechanisms of a recommender system, for example, by giving preference to recommendations from sources that are trusted more, or to opinions that are better informed. An example of the former approach can be found in our new distrust-aware recommendation strategy of Section 3.

In Victor et al. [2009a], more background information on bilattice-based trust and distrust modeling can be found, along with a classification of trust models and additional examples.

2.1. Trust Score Propagation Operators

In virtual trust networks, propagation operators are used to handle the problem of establishing trust information in an unknown user by inquiring through other users. The simplest case, atomic propagation, takes the trust score $(t_{a,b}, d_{a,b})$ of user a in user b and the trust score $(t_{b,x}, d_{b,x})$ of b in user x, and uses this information to predict the trust score of a in x. In Victor et al. [2009a], four operators were proposed for this purpose, each reflecting a different strategy for dealing with the available distrust information. In this article, we focus on the propagation operator that makes the most active use of distrust information.

To study the propagation scheme, let us first consider the bivalent case, that is, when trust and distrust degrees assume only the values 0 or 1. For agent a and agent b, we use $t_{a,b}$ and $d_{a,b}$ as shorthands for respectively $t_{a,b} = 1$ and $d_{a,b} = 1$. Let us consider the following propagation scheme:

$$t_{a,x} \equiv (t_{a,b} \wedge t_{b,x}) \vee (d_{a,b} \wedge d_{b,x}) \text{ and } d_{a,x} \equiv (t_{a,b} \wedge d_{b,x}) \vee (d_{a,b} \wedge t_{b,x}).$$

An agent a exhibiting this behavior listens to whom he trusts by copying the opinion of the trusted third party: a will trust x if a trusts b and b trusts x, while a will distrust x if a trusts b and b distrusts x. Furthermore, the distrust-aware part of the propagation operator corresponds to an interpretation in which the enemy of an enemy is considered to be a friend, and the friend of an enemy to be an enemy too: a will also trust x if a distrusts b and b distrusts x, and a will also distrust x if a distrusts b and b trusts x.

In a trust score setting, besides 0 and 1, we also allow partial trust and distrust. Hence we need suitable extensions of the logical operators that are used in the preceding scheme. For conjunction and disjunction, we use respectively a t-norm and a t-conorm [Schweizer and Klar 1961]. We use $\mathcal{T}$ to denote an arbitrary t–norm, that is, an increasing, commutative, and associative $[0, 1]^2 \to [0, 1]$ mapping satisfying $\mathcal{T}(1, x) = x$ for all x in $[0, 1]$. Furthermore, $\mathcal{S}$ denotes an arbitrary t–conorm, that

Table I. Examples of t-Norms and t-Conorms, with x and y in $[0, 1]$

t-norms	t-conorms
$T_{\mathrm{M}}(x, y) = \min(x, y)$	$S_{\mathrm{M}}(x, y) = \max(x, y)$
$T_{\mathrm{P}}(x, y) = x \cdot y$	$S_{\mathrm{P}}(x, y) = x + y - x \cdot y$
$T_{\mathrm{L}}(x, y) = \max(x + y - 1, 0)$	$S_{\mathrm{L}}(x, y) = \min(x + y, 1)$

is, an increasing, commutative, and associative $[0, 1]^2 \rightarrow [0, 1]$ mapping satisfying $\mathcal{S}(0, x) = x$ for all x in $[0, 1]$. T-norms and t-conorms represent large classes of logic connectives, from which specific operators, each with its own behavior, can be chosen, according to the application or context. Table I contains some well-known and often used t-norms and t-conorms.

In the remainder of Section 2, we use t_1 as an abbreviation for the trust degree $t_{a,b}$ of agent a in agent b, and d_1 for the corresponding distrust degree $d_{a,b}$. Similarly, we use (t_2, d_2) to denote the trust score from agent b in agent x. In other words, the trust score of a in b is denoted by (t_1, d_1) and the one for b in x as (t_2, d_2). The extension of the bivalent propagation scheme leads to the following distrust-aware propagation operator.

Definition 2.3 (*Distrust-Enhanced Propagator*). [Victor et al. 2009a]. Let $\mathcal{T}$ be a t-norm and $\mathcal{S}$ a t-conorm. The propagation operator P is defined by (for (t_1, d_1) and (t_2, d_2) in $[0, 1]^2$):

$$P((t_1, d_1), (t_2, d_2)) = (\mathcal{S}(\mathcal{T}(t_1, t_2), \mathcal{T}(d_1, d_2)), \mathcal{S}(\mathcal{T}(t_1, d_2), \mathcal{T}(d_1, t_2))).$$

Since P is not associative, we have to fix a propagation order when we want to establish a link between a and x using more intermediate third parties. Here, we assume that a right-to-left evaluation order (backward propagation) is used, that is, we recursively define, for $m > 2$,

$$P^m((t_1, d_1), \ldots, (t_m, d_m)) = P^2((t_1, d_1), P^{m-1}((t_2, d_2), \ldots, (t_m, d_m))).$$

Note that P is a distrust intensive operator, because a distrusted acquaintance can play a role in the determination of both the trust and distrust degree of the propagated trust score: d_1 appears in the first as well as the second argument in the right side of the formula. Experiments in Victor et al. [2011c] on the Epinions dataset used in Section 4 have shown that, compared to other propagation operators, P is able to achieve lower prediction errors on average. Moreover, recent studies on social datasets have revealed that several types of trust-based applications may benefit from such a distrust-enhanced operator. For example, it has been shown that "the enemy of an enemy is a friend" propagation pattern (in line with P) is applicable in the technology news Web site Slashdot [Kunegis et al. 2009] and the political forum Essembly [Hogg et al. 2008].

2.2. Trust Score Aggregation Operators

When a user a needs to establish an opinion about another user x, and there is more than one path linking them, we require a way of combining the information provided by each of those paths. This is where aggregation operators come into play. In Victor et al. [2010], we postulated three desirable properties for a trust score aggregation operator. These are illustrated in Figure 1, where eight trust scores, represented by dots, have

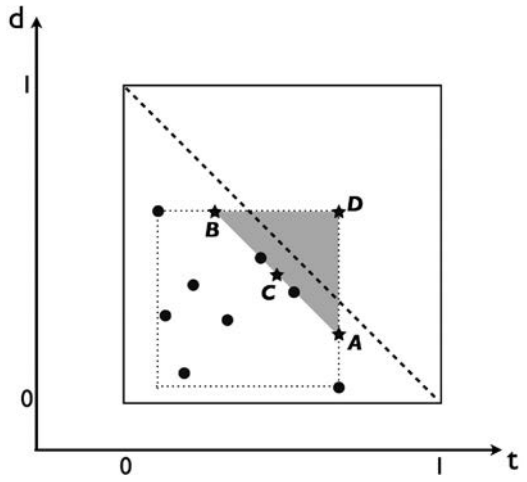

Fig. 1. Example of eight trust scores (dots) to be aggregated.

to be aggregated. First, the trust and distrust boundary conditions ensure that the aggregated trust score reflects a consensus about the (dis)trust estimation: it should not contain more (dis)trust than the maximum (dis)trust value among the aggregates (and analogously for the minimum). The trust and distrust boundaries yield a limited number of possible aggregation results, marked out by the dotted lines. Furthermore, the so-called knowledge boundary condition ensures that the knowledge contained in the aggregated trust score (t, d) (i.e., $t + d$) does not decrease when aggregating additional opinions. In other words, the aggregated trust score will have a knowledge degree that is at least as high as that of the most knowledgeable aggregate. By also imposing the knowledge condition, only part of the possibilities remain, depicted by the gray area in Figure 1.

Each of the trust scores marked by stars make sense as aggregated score: A is the most optimistic choice (maximum trust degree t for the lowest possible knowledge level), B the most pessimistic one (maximum distrust degree d), C the moderating approach (average of the most knowledgeable scores), and D the most extreme, knowledge maximizing, option: maximum t and d, often resulting in an inconsistent trust estimation. These strategies lead to the following four trust score aggregation operators, respectively.

Definition 2.4 (*TMAX*). [Victor et al. 2010]. The trust maximizing aggregation operator *TMAX* is defined by (for $(t_1, d_1), \ldots, (t_n, d_n)$ in $[0, 1]^2$)

$$TMAX\left((t_1, d_1), \ldots, (t_n, d_n)\right) = (p, q)$$

with (p, q) such that

$$p = \max(t_1, \ldots, t_n), \quad q = \max(t_1 + d_1, \ldots, t_n + d_n) - \max(t_1, \ldots, t_n).$$

Definition 2.5 (*DMAX*). [Victor et al. 2010]. The distrust maximizing aggregation operator *DMAX* is defined by (for $(t_1, d_1), \ldots, (t_n, d_n)$ in $[0, 1]^2$)

$$DMAX\left((t_1, d_1), \ldots, (t_n, d_n)\right) = (p, q)$$

with (p, q) such that

$$p = \max(t_1 + d_1, \ldots, t_n + d_n) - \max(d_1, \ldots, d_n), \quad q = \max(d_1, \ldots, d_n).$$

Definition 2.6 (*KAV*). [Victor et al. 2010]. The knowledge preference averaging aggregation operator *KAV* is defined by (for $(t_1, d_1), \ldots, (t_n, d_n)$ in $[0, 1]^2$)

$$KAV\left((t_1, d_1), \ldots, (t_n, d_n)\right) = (p, q)$$

with (p, q) such that

$$p = \frac{\sum_{i=1}^{n} w_i \cdot t_i}{\sum_{i=1}^{n} w_i}, \quad q = \frac{\sum_{i=1}^{n} w_i \cdot d_i}{\sum_{i=1}^{n} w_i}$$
$$w_i = \begin{cases} 1 & \text{if } t_i + d_i = \max(t_1 + d_1, \ldots, t_n + d_n). \\ 0 & \text{otherwise} \end{cases}$$

Definition 2.7 (*KMAX*). [Victor et al. 2010]. The knowledge maximizing aggregation operator *KMAX* is defined by (for $(t_1, d_1), \ldots, (t_n, d_n)$ in $[0, 1]^2$)

$$KMAX\left((t_1, d_1), \ldots, (t_n, d_n)\right) = (p, q)$$

with (p, q) such that

$$p = \max(t_1, \ldots, t_n), \quad q = \max(d_1, \ldots, d_n).$$

All these operators actively take into account trust and distrust and can hence be used in distrust-enhanced recommendation strategies, as we will discuss in the following section.

3. DISTRUST-ENHANCED RECOMMENDATIONS

Although the area of trust-enhanced recommendations is still a very young research domain, already quite a few approaches have been proposed. We refer to Golbeck [2009] or Victor et al. [2011b] for an overview of classic and novel contributions in the field of trust-aware recommender systems. In the following section, we focus on one of the best known trust-enhanced recommendation strategies, namely Golbeck's approach. We explain the basics of her method and then continue by exploring ways to enhance the algorithm by incorporating distrust. We propose a new distrust-based recommender approach in Section 3.2 and discuss related work in Section 3.3.

3.1. Trust-Based Weighted Mean (TBWM)

In a recommender system without a trust network, a simple recommendation algorithm that needs to estimate how well a target user a will like a target item i can compute the average of all ratings $r_{u,i}$ for i from all the system's users u who are already familiar with i. This baseline recommendation strategy can be refined by computing a trust-based weighted mean. In particular, by including trust values $t_{a,u}$ (direct or inferred through a trust metric) that reflect the degree to which the raters u are trusted, the algorithm allows to differentiate between the sources. In fact, it is only natural to assign more weight to ratings of highly trusted users; the formula is given in Definition 3.1.

Definition 3.1 (*Trust-Based Weighted Mean (TBWM)*). [Golbeck 2005]. The unknown rating for target item i and target user a can be computed as

$$p_{a,i} = \frac{\sum_{u \in R^T} \hat{t}_{a,u} r_{u,i}}{\sum_{u \in R^T} \hat{t}_{a,u}},$$

with R^T the set of users who evaluated i and for whom the trust value $\hat{t}_{a,u}$ is greater than or equal to a threshold $\alpha > 0$ (i.e., neighbors of a). The trust value $\hat{t}_{a,u}$ equals $t_{a,u}$ if the trust information of a in u is directly available, or is estimated using a trust metric otherwise.

The formula in Definition 3.1 is at the heart of Golbeck's recommendation algorithm [Golbeck 2005]. The novelty of the algorithm mainly lies in the way the trust estimates $\hat{t}_{a,u}$ are inferred, by means of a trust metric that she called TidalTrust. TidalTrust only takes into account the shortest and strongest trust paths to the target item. More specifically, the shortest path length that is needed to connect the target user with a user u who has rated the target item (i.e., a rater) becomes the path depth of the algorithm. Furthermore, Golbeck opted to incorporate a value that represents the path strength (i.e., the minimum trust rating on a path leading to the user who is connected with u), and to compute the maximum path strength over all paths leading to the raters. This maximum (*max*) is then chosen as the minimum trust threshold for participation in the process. In other words, only users for whom the trust value is greater than or equal to the dynamic path strength threshold *max* can participate in the process.

Definition 3.2 (*TidalTrust*). [Golbeck 2005]. The trust value from target user a in user u is estimated recursively as

$$\hat{t}_{a,u} = \frac{\sum_{v \in WOT^+(a)} t_{a,v} \cdot \hat{t}_{v,u}}{\sum_{v \in WOT^+(a)} t_{a,v}},$$

with $WOT^+(a)$ the set of users in the personal web of trust of user a (which we denote by $WOT(a)$) for whom a's trust statement is greater than or equal to the dynamically computed threshold *max*. If $WOT^+(a)$ is empty, then $\hat{t}_{a,u} = 0$.

3.2. Trust Score Based Weighted Mean (TSBWM)

The aforesaid algorithms are specifically designed for a trust-only environment. In this section, however, we explore how distrust may enhance the trust-based recommendation process. Since now we have to deal with distrust, we require adapted propagators, such as the propagation operator from Definition 2.3. Whenever trust score aggregation operators are needed, all operators from Section 2.2 can be used.

The utility of distrust for recommendation algorithms can be explored in several ways. One commonly suggested (but not previously evaluated) approach is that distrust be used to debug a web of trust (see, e.g., Guha et al. [2004], and Ziegler and Lausen [2005]): suppose that a trusts b completely, b fully trusts x, and a completely distrusts x, then the latter fact invalidates the propagated trust result (viz. a trusts x). As such, distrust-enhanced algorithms may be useful to filter out "false positives" in the propagated web of trust of a target user. Consequently, a user who would otherwise participate will now be excluded from the recommendation process. Such a debug strategy can be implemented in several ways; in the following definition we present a new distrust-enhanced recommendation approach which is an adaptation of Golbeck's trust-based weighted mean.

Definition 3.3 (*Trust Score-Based Weighted Mean (TSBWM)*). The unknown rating for target item i and target user a can be computed as

$$p_{a,i} = \frac{\sum_{u \in R} \max\left(0, \hat{t}_{a,u} - \hat{d}_{a,u}\right) \cdot r_{u,i}}{\sum_{u \in R} \max\left(0, \hat{t}_{a,u} - \hat{d}_{a,u}\right)},$$

with R the set of users who evaluated i. The trust score $(\hat{t}_{a,u}, \hat{d}_{a,u})$ denotes a's direct evaluation of u, namely, $(t_{a,u}, d_{a,u})$, or the trust score estimation inferred through a trust metric otherwise.

In the scenario of Definition 3.3, we use the trust scores as a way to create the weights for neighbors, and at the same time also as a filter for the neighbors: the more a neighbor is trusted and the less he is distrusted, the higher his weight in the aggregation process. In other words, neighbors are rewarded according to their trust/distrust difference. Furthermore, users for whom there is more evidence to distrust than to trust are filtered out.

Since now also distrust values are involved, we need a new trust metric: as explained in the previous section, working with two degrees (i.e., a trust score) requires new propagation and aggregation strategies. Therefore, we propose a new family of trust metrics that compute one trust score estimation instead of a separate trust and distrust estimation. Similar to Golbeck's approach, we only take into account the shortest and strongest paths to the target item i. The threshold max_t is computed analogously to Definition 3.2, and min_d as the minimum of the distrust path strengths (the maximum distrust rating on a path).

Definition 3.4 (*Distrust-Aware Trust Metric Family*). The trust score for target user a in user u can be computed by

$$(\hat{t}_{a,u}, \hat{d}_{a,u}) = \underset{i=1\cdots m}{A} \left(P\left((t_1, d_1)_i, \ldots, (t_n, d_n)_i\right)\right),$$

in which $(t_1, d_1)_i, \ldots, (t_n, d_n)_i$ denotes the ith trust score path (of length $n > 1$) from a to u, A is taken to be an aggregator from Section 2.2 and P the propagator from Definition 2.3. Each participating trust score (t_j, d_j) (with $j = 1 \cdots n$) must be greater than or equal to the dynamically computed threshold (max_t, min_d), that is, $(t_j, d_j) \geq_{td} (max_t, min_d)$, or in other words, $t_j \geq max_t$ and $d_j \leq min_d$. If no paths can be found, then $(\hat{t}_{a,u}, \hat{d}_{a,u}) = (0, 0)$.

Note that the implementation of this approach differs from TidalTrust since it uses the "first propagate then aggregate" strategy, while the latter follows the "first aggregate then propagate" strategy. In Definition 3.4, first all paths to the target item are looked up, then for each of them a propagation operator is applied, and finally all propagated trust scores are aggregated. On the other hand, TidalTrust is a recursive approach in which the aggregation process does not take place once, but at every point where two propagation chains meet.

3.3. Related Work

In the area of trust-enhanced recommendation algorithms, besides approaches using a weighted mean, there also exist methods that are based on the classic collaborative filtering algorithm [Resnick et al. 1994], in which the unknown rating for the target item i is computed as a combination of the deviations of ratings for i by users u (with respect to u's average rating behavior). Only users u whose rating behavior is correlated to that of the target user a are taken into account.

O'Donovan and Smyth's trust-based filtering [O'Donovan and Smyth 2005] is tied very closely to the collaborative filtering algorithm: it adapts the latter by only taking

into account trustworthy neighbors, that is, users u who are trusted by, and have a positive correlation with, the target user. Like this, the trust information is used to filter the set of possible neighbors.

Instead of a correlation-based computation of the weights, one can also infer the weights through the relations of the target user in the trust network, as in Golbeck's approach. This results in an adaptation of the collaborative filtering formula in which the correlation-based weights are replaced by the trust values $\hat{t}_{a,u}$. This strategy is at the heart of Massa and Avesani's trust-based collaborative filtering; see Massa and Avesani [2009].

The preceding algorithms can serve as a basis for new distrust-enhanced recommendation algorithms. In Victor et al. [2011d] we reflected a first time upon the possible roles of distrust. One possibility we explored was to use distrust as an indicator to reverse deviations, that is, by considering distrust scores as negative weights, analogous to the use of negative correlation coefficients in a classic collaborative filtering approach, or to the model-based approach in Ma et al. [2009], which also uses distrust to denote dissimilar users. However, experiments in Victor et al. [2011c] showed that this is not the direction to take for memory-based approaches.

Another approach looked more promising, namely to utilize distrust as a filter, analogous to the filter in O'Donovan and Smyth's [2005] approach. Like this, distrust evidence can be used to leave out "unwanted" individuals from the recommendation process: instead of using all users similar to the target user in a collaborative filtering process, one can also restrict the set of neighbors such that only similar users who are not distrusted by the target user (or for whom the distrust estimation does not exceed a certain threshold) are taken into account.

Analogously, the set of users that participate in the recommendation process in Golbeck's strategy (Definition 3.1) can also be restricted by only retaining the users for which there does not exist evidence that they should be distrusted (to some degree). This led to the following algorithm.

Definition 3.5 (*Debugged Trust-Based Weighted Mean (DTBWM)*). [Victor et al. 2011a]. The unknown rating for target item i and target user a can be computed as

$$p_{a,i} = \frac{\sum_{u \in R^{T+}} \hat{t}_{a,u} r_{u,i}}{\sum_{u \in R^{T+}} \hat{t}_{a,u}},$$

with R^{T+} the set of users who evaluated i, for whom the trust value $\hat{t}_{a,u}$ is greater than or equal to a given threshold α and for whom the distrust value $\hat{d}_{a,u}$ equals to zero. The distrust value $\hat{d}_{a,u}$ equals $d_{a,u}$ if the distrust information of a in u is directly available, or is estimated using a trust metric otherwise.

To ensure that debugged trust-based weighted mean behaves the same way as trust-based weighted mean when no distrust is involved, the trust estimation $\hat{t}_{a,u}$ is computed as in TidalTrust (Definition 3.2). The distrust estimation is obtained by using Definition 3.4 with $A = DMAX$ and P as in Definition 2.3. The propagation operator follows the "distrust your enemy's friends, as well as your friend's enemies" pattern; in other words, the set of distrusted users contains all users who are directly distrusted by the target user a, who are distrusted by the members of a's WOT, users in the WOT of users who are distrusted by a, etc. By using *DMAX* as aggregator, one ensures that $\hat{d}_{a,u}$ will be greater than zero as soon as there is at least one path to u which results in distrust information, since *DMAX* maximizes distrust. Consequently, Definition 3.5 is a strong debug implementation: whenever there is any trace of distrust evidence, u is excluded from the recommendation process.

Remark that *KMAX* yields the same results, but that *TMAX* or *KAV* are less suited to implement this particular debug strategy, since they will not always result in $\hat{d}_{a,u} \neq 0$ when there is distrust evidence present: the former maximizes trust (and hence results in $\hat{d}_{a,u} = 0$ whenever there is a propagated trust score that denotes complete trust), while the latter only takes into account the most knowledgeable opinions (and hence ignores all others, which may possibly contain distrust information).

4. EXPERIMENTS AND DISCUSSION

Since the goal of this article is to experimentally evaluate whether incorporating distrust can indeed enhance the trust-based recommendation process, we compare the performance of Golbeck's trust-based weighted mean (Definition 3.1) with its debugged counterpart (Definition 3.5) and trust-score-based weighted mean (Definition 3.3) for the controversial items in Epinions. Because the latter two algorithms require a distrust-aware trust metric, we need to choose a specific implementation of Definition 3.4; we will experiment with several aggregators so that we can investigate the impact of the aggregation choice on the accuracy and coverage of the recommendations.

4.1. Methodology

The dataset we use in our experiments is obtained from Epinions.com, a popular e-commerce site where users can write reviews about consumer products and assign a rating to the products and the reviews. The reviews are evaluated by assigning a helpfulness rating which ranges from "not helpful" (1/5) to "most helpful" (5/5). The dataset, compiled by Guha et al. [2004], does not contain any information about consumer products and product ratings, but works with reviews and review ratings instead; in other words, we will discuss and evaluate a "review recommender system". Hence, in this context, an item denotes a review of consumer goods.

In our experiments we focus on the number of recommendations/predictions that can be generated by the system and on the prediction errors for controversial items (see the following). These are the most challenging items for a recommender system, since it is much harder to predict a score for an item that has received a variety of high and low scores, reflecting disagreement about the item. More than in any other case, a recommendation for a user needs to be truly personalized when the target item under consideration is controversial; that is, when an item has both "ardent supporters" and "motivated adversaries", with no clear majority in either group. In Victor et al. [2009b], we have identified 1 416 of such controversial items in Guha et al.'s dataset which we use in our experiments that follow. Furthermore, in order to assess the performance of the recommendation strategies in a general, noncontroversial setting, we have also selected 1 416 random items for our experiments.

Epinions allows users to evaluate other users based on the quality of their reviews, and to provide trust and distrust evaluations in addition to ratings. The fact that the dataset contains explicit trust and distrust information from the users makes it very appropriate to study issues in trust-enhanced recommender systems. Users can evaluate other users by including them in their WOT (i.e., a list of reviewers whose reviews and ratings were consistently found to be valuable[4]), or by putting them in their block list (a list of authors whose reviews were consistently found to be offensive, inaccurate, or low quality[4], thus indicating distrust). In the dataset, the trust evaluations make up an Epinions WOT graph consisting of 114 222 users and 717 129 nonself-referring trust relations; about 85% of all statements denote trust.

Note that the dataset only contains bivalent trust values, hence in our experiments $t_{a,u}$ and $d_{a,u}$ in Definitions 3.1, 3.3, and 3.5 can take on the values 0 (absence of trust)

[4]See www.epinions.com/help/faq/.

and 1 (full presence) only. This limitation leads to alterations of some of the algorithms; for example, the formula in Definition 3.1 reduces to the classical average computed over all trusted users. With respect to Definition 3.5, also note that by choosing $A = TMAX$, the formula reduces to the formula in Definition 3.1: any trust evidence yields $(\hat{t}_{a,u}, \hat{d}_{a,u}) = (1, 0)$ due to the bivalent nature of the trust scores, and hence any trusted user will take part in the recommendation process. Finally, the value of the parameter α in Definition 3.5 is of little consequence, as any value greater than 0 will result in the same outcome.

To measure the performance of the recommendation algorithms of Definitions 3.1, 3.3, and 3.5 we use the leave-one-out method which consists of hiding a rating and trying to predict its hidden value. In particular, we use two well-known accuracy measures, namely, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) [Herlocker et al. 2004]. The first measure considers every error of equal value, while the latter one emphasizes larger errors. Since reviews are rated on a scale from 1 to 5, the extreme values that MAE and RMSE can reach are 0 and 4. Even small improvements in RMSE are considered valuable in the context of recommender systems. For example, the Netflix prize competition[5] offered a \$1 000 000 reward for a reduction of the RMSE by 10%.

Besides accuracy, we also consider the coverage that a recommender algorithm can achieve, that is, the number of target user-target item pairs for which a prediction can be generated. In a leave-one-out context, the coverage of a specific algorithm then refers to the amount of computable predictions $p_{a,i}$ versus the number of leave-one-out experiments to perform (the number of ratings available in the dataset).

The algorithms in Definitions 3.1, 3.3, and 3.5 only take into account the shortest paths to the target item. Sometimes an item will immediately be found because we have direct trust information from the target user to a user who has rated the target item (the rater). In this case, we say that the item is found on level 1 (L1). It is also possible that the shortest path to the target item is a chain length of 2, meaning that the target user and the rater are connected via one trusted third party, and that the trust information needs to be propagated. In other words, in this case, the item can be found on level 2 (L2). Obviously, it will also happen that longer propagation chains are required to reach the target item.

In our experiments, we also want to study the effect of propagation on the recommendations. To this aim, we first limit the radius around the target user to L1, meaning that only direct trust information can be used. Subsequently we relax up to L2, and then up to L3. For a L3 experiment, for example, this means that the shortest paths to the target items can consist of one link, two, or three links (of course, for one particular target item, all shortest paths have the same length). For the propagation operator, the choice of t-norm and t-conorm does not really matter in our experiments because the dataset only contains bivalent trust and distrust statements, and hence all t-norms and t-conorms behave in the same way by definition.

4.2. Results

Table II contains the results of our experiments for controversial items. The numbers in the first column refer to the formulas in Section 3. Note that the numbers in the L1 column are the same for each algorithm: both Definitions 3.3 and 3.5 use distrust as a debugger of a web of trust, and since the Epinions dataset does not contain inconsistent information (where a user can simultaneously trust and distrust another user), debugging only makes sense for propagated information.

[5]See http://www.netflixprize.com/.

Table II. Performance of Distrust-Based Algorithms on Epinions' Controversial Items, with $\mathcal{T} = T_{\mathrm{M}}$ and $\mathcal{S} = S_{\mathrm{M}}$

DEF	*ALGORITHM*	*L1*			*L2*			*L3*		
		% COV	MAE	RMSE	% COV	MAE	RMSE	% COV	MAE	RMSE
3.1	TBWM [Golbeck 2005]	63	0.86	1.20	88	0.91	1.22	90	0.93	1.24
3.5	DTBWM–TMAX	63	0.86	1.20	88	0.91	1.22	90	0.93	1.24
3.5	DTBWM–KAV [Victor et al. 2011d]	63	0.86	1.20	86	0.91	1.23	64	0.86	1.20
3.3	TSBWM–TMAX	63	0.86	1.20	86	0.86	1.17	87	0.87	1.18
3.3	TSBWM–KAV	63	0.86	1.20	86	0.85	1.17	87	0.86	1.17

4.2.1. Trust-Based Weighted Mean. Let us first concentrate on the first row of the table, to discuss the effect of trust propagation on the trust-only approach trust-based weighted mean. One can immediately notice the significant coverage benefit that is gained by propagating trust information. The profit is especially high for the transition from level 1 (direct information) to level 2, afterwards the bonus becomes less, but by then already a coverage of 90% is reached. However, the downside of using propagation is that the accuracy of the recommendations decreases: the longer the allowed propagation chains, the further away we are heading from the target user, and hence the less accurate the trust predictions, which will also affect the accuracy of the recommendations. This observation is in line with the results reported in Golbeck [2005] and Massa and Avesani [2009].

4.2.2. Debugged versus Original Trust-Based Weighted Mean. In the second and third row of Table II we focus on the utility of distrust as a debugger for a target user's web of trust. This results in Definition 3.5, an extended version of Definition 3.1. Remark that choosing *DMAX* or *KAV* or *KMAX* as aggregation operator yields the same results for the debugger type in Definition 3.5 due to the bivalent nature of the data: whenever there is any distrust evidence, $\hat{d}_{a,u} \neq 0$, and hence the user will not take part in the recommendation process. Recall that the implementation of Definition 3.5 with $A = TMAX$ generates the same results as Definition 3.1; in the remainder of this section, we focus on the performance of debugged trust-based weighted mean with $A = KAV$.

On level 2, the strategy leads to a coverage decrease of 2% for the debugged version of trust-based weighted mean compared to its original propagated counterpart (Definition 3.1). In other words, using one-step distrust propagation to filter out false positives results in an almost unchanged accuracy, and only has a marginal effect on the coverage. This can be explained by the trust/distrust ratio in the dataset (recall that merely about 15% of all relations denote distrust) and the fact that the controversial items are also popular (meaning that they received a lot of ratings), hence there is often at least one neighbor to participate in the recommendation process (so that a recommendation can be generated).

However, once we start propagating one step further (L3), the impact of the debugger becomes much more visible: the MAE and RSME decrease, but along with it also the coverage. The latter even worsens significantly: compared to its propagated non-debugged counterpart, the coverage decreases with 26%. This tells us that the kind of debugger of Definition 3.5 is too extreme for the Epinions application; the longer the propagation paths that we take into account, the more often distrust evidence can be found for a particular user, and hence the less (or sometimes no) neighbors will be left to participate in the recommendation process.

4.2.3. Trust-Score-Based versus Debugged Trust-Based Weighted Mean. As discussed in Section 3.2, another way to use distrust as a debugger is to integrate it in the

determination of the weights, as in our new algorithm trust-score-based weighted mean (Definition 3.3). In the bottom part of the table, we experiment with two implementations of Definition 3.4. In particular, we tested the aggregation operators from Section 2.2. Note that KMAX and DMAX are not included because the dataset only contains bivalent trust and distrust data, and hence $\max(0, \hat{t}_{a,u} - \hat{d}_{a,u})$ will always yield weight 0 whenever there is at least 1 distrust input; consequently few users will be able to take part in the recommendation process, and hence almost no recommendations will be generated. Recall that the results for the unpropagated version are the same as the ones from Definition 3.5.

Let us compare debugged trust-based weighted mean (Definition 3.5) with trust-score-based weighted mean (Definition 3.3). With regard to coverage on level 2, the three strategies perform almost equally well. TMAX yields somewhat more recommendations than KAV: the former results more often in $t_{a,u} - d_{a,u} > 0$ (due to its optimistic nature, i.e., trust maximizing behavior) than in ≤ 0 (which can occur when all inputs denote full trust or ignorance; a rare scenario). While the coverages remain comparable, the accuracy clearly improves when changing the debug strategy. As far as the mutual accuracy relations between the implementations of Definition 3.3 are concerned, there is no significant difference; the KAV implementation performs slightly better with respect to MAE.

On level 3, however, the picture looks completely different, especially with respect to coverage. Whereas the strong debugging of Definition 3.5 caused a great coverage loss when two-step propagation is taken into account, the weight-based debugger adjusts itself much better to longer propagation chains: more recommendations can be made on L3 compared to L2 due to the moderating behavior of the debug strategy. The accuracy of debugged trust-based weighted mean and trust-score-based weighted mean on L3 is more or less the same, with a small improvement on MAE and RMSE for the latter. Hence, taking into account the much higher coverage that trust-score-based weighted mean can achieve, it is fair to state that trust-enhanced recommendation algorithms that incorporate a debugging method benefit the most from less marked debug implementations, and hence that trust-score-based weighted mean outperforms the debugged trust-based weighted mean approach.

4.2.4. Trust-Score-Based versus Original Trust-Based Weighted Mean. Let us now compare the performance of trust-based and trust-score-based weighted mean (Definitions 3.1 and 3.3). Obviously, when using a debug strategy, the coverage of any trust-enhanced algorithm will always decrease compared to its trust-only counterpart, but Table II shows us that the coverage certainly remains acceptable, with only a 2% and 3% loss on respectively level 2 and 3. With respect to accuracy, trust-score-based weighted mean clearly improves trust-based weighted mean, on L2 as well as L3, and for both MAE and RMSE.

To demonstrate that the accuracy increase cannot entirely be attributed to the coverage-accuracy trade-off (increases in coverage are often at the expense of accuracy, and vice versa), we also performed a second set of experiments. We followed the same procedure as in the first experiment but used particular subsets of the controversial items, namely the intersection of the controversial items that can be recommended by Definition 3.1 and the controversial items that can be recommended by each implementation of Definition 3.3. Like this, all algorithms can achieve the same coverage. Table III shows the results for the common controversial items that can be reached by one-step propagation (L2) and two-step propagation (L3).

We also performed the Wilcoxon signed rank test [Wilcoxon 1945] to verify whether the differences obtained for MAE in Table III are statistically significant. For each comparison, in Table IV, we show R^+, the mean of positive rankings and R^-, the

Table III. Performance of the Algorithms on the Same Subset of Controversial Items on L2 and L3

DEF	*ALGORITHM*	*L2*		*L3*	
		MAE	RMSE	MAE	RMSE
3.1	TBWM	0.91	1.23	0.92	1.24
3.3	TSBWM–TMAX	0.88	1.18	0.87	1.18
3.3	TSBWM–KAV	0.88	1.18	0.86	1.18

Table IV. Results of the Signed-Rank Wilcoxon Test for the MAE Results in Table III

COMPARISON	*L2*			*L3*		
	R^+	R^-	p-value	R^+	R^-	p-value
TBWM vs. TSBWM–TMAX	8407.57	9722.18	<0.0001	8886.92	9860.68	<0.0001
TBWM vs. TSBWM–KAV	8807.91	8933.90	<0.0001	9017.49	9464.52	<0.0001
TSBWM–TMAX vs. TSBWM–KAV	2471.24	2976.17	<0.0001	2621.94	3112.21	<0.0001

Table V. Performance of Distrust-Based Algorithms on Epinions' Random Items, with $\mathcal{T} = T_{\mathrm{M}}$ and $\mathcal{S} = S_{\mathrm{M}}$

DEF	*ALGORITHM*	*L1*			*L2*			*L3*		
		% COV	MAE	RMSE	% COV	MAE	RMSE	% COV	MAE	RMSE
3.1	TBWM	87	0.133	0.355	97	0.147	0.381	97	0.150	0.388
3.5	DTBWM–TMAX	87	0.133	0.355	97	0.147	0.381	97	0.150	0.388
3.5	DTBWM–KAV	87	0.133	0.355	96	0.145	0.379	88	0.134	0.358
3.3	TSBWM–TMAX	87	0.133	0.355	96	0.147	0.381	96	0.149	0.386
3.3	TSBWM–KAV	87	0.133	0.355	96	0.147	0.380	96	0.148	0.386

mean of negative rankings, and the corresponding p-value[6] when a 5% significance level is used. It is clear that in each of the comparisons, the differences are significant according to the test, that is: TSBWM–KAV outperforms TSBWM–TMAX, which in turn outperforms TBWM.

These results clearly reinforce our claim that actively involving distrust in the trust-enhanced recommendation process is beneficial for the quality of the recommendations, and that using distrust as a debugger can improve the performance of trust-only approaches that incorporate trust propagation techniques; in particular, trust-score-based weighted mean outperforms trust-based weighted mean.

4.2.5. Comparison for Random Items. Looking at the results in Table V, it can immediately be noticed that for random items, prediction errors are much lower than for controversial ones. Moreover, coverage is higher since a random item typically gets rated more frequently than a controversial one.

When comparing the various trust-enhanced strategies, it can be seen that their differences are much smaller than in the previous case. Prediction errors are similar for all strategies, but debugged trust-based weighted mean still suffers the problem that on L3, its coverage is almost 10% lower than that of the other strategies. Trust-score-based weighted mean, on the other hand, obtains almost the same coverage as trust-based weighted mean, while it also maintains the same accuracy.

[6]The p-value was calculated based on the asymptotic normality of the Wilcoxon test statistic, using release 19.0.0.1 of the statistical package SPSS.

Concluding, we can state that for general, random items, trust-score-based weighted mean and trust-based weighted mean are comparable, while for controversial items, the former presents a real advantage over the latter.

4.2.6. Performance Analysis. The trust metrics using distrust do not require significantly more computations than the basic ones. On the other hand, incorporating debugging strategies results in a lower coverage, hence, it is more often necessary to look for paths on a higher level.

The difference between the trust-based weighted mean and debugged trust-based weighted mean algorithm is that the latter has a stronger restriction for using a user's rating for an item. That is, the rating is only used if the user is not distrusted. It can occur that the trust-based weighted mean can provide a predicted rating for a certain level, but that the debugged trust-based weighted mean strategy has to look for a prediction on a higher level. As a result, the debugged trust-based weighted mean algorithm will take longer.

The same holds for the trust-score-based weighted mean strategy. By using the adjusted version of the TidalTrust algorithm that also takes into account the minimum distrust strength, it can occur that the original TidalTrust algorithm finds paths between the two users at hand on a lower level than the adjusted TidalTrust algorithm. As a result, it will take longer than the trust-score-based weighted mean strategy.

Note that the trust-score-based mean debugging strategy is less strict than the debugged trust-based weighted mean strategy. As a result, the latter will more often need to find paths on a higher level and will take longer than the trust-score-based mean debugging method.

5. CONCLUSIONS AND FUTURE WORK

With the growing popularity of online social networks, trust-enhanced recommendations techniques will become an asset for future generations of Web applications. However, in large user communities, it is only natural that besides trust also distrust starts to emerge. Hence, the more users issuing distrust statements, the more interesting it becomes to also incorporate this new information source.

In this article, we embarked upon the distrust-enhanced recommendation problem, a research area that is still in its very infancy. We experimentally investigated the potential of distrust as a debugger of the users' propagated web of trust, and the necessary changes that such a strategy brings along for the trust metric implementation. We showed that debugging methods must exhibit a moderate behavior in order to be effective.

We proposed a new algorithm, trust-score-based weighted mean, which is a distrust-enhanced extension of Golbeck's approach. Through experiments on a dataset from the e-commerce site Epinions, we demonstrated that, if the trust metric implementation is carefully chosen, distrust-aware recommendation algorithms can outperform their trust-only counterparts for controversial items: more accurate recommendations can be obtained without a significant loss in coverage; the results clearly show that distrust information can indeed be beneficial for the recommendation process.

In our experiments we mainly focused on the influence of the aggregation operator on the quality and quantity of distrust-aware recommendations. However, the propagation operator is also a determining factor. Hence, our future research first involves the investigation of the influence of propagators and the synergy between the two operator types. In a next step, we want to further examine which types of debugger deliver the best results, and to focus on the enhancement of other algorithm classes; think, for example, of debugged versions of collaborative filtering-based trust-aware approaches, either memory based or model based.

Another important challenge is the generation of the kind of data that would foster more comprehensive experimental studies. As we explained in the introduction to our article, the Epinions dataset contains only bivalent trust information, and to our knowledge there currently exist no publicly available datasets that include gradual trust and distrust information for the recommendation task. One possible avenue to tackle this problem is the generation of adequate synthetic data to test our approach. Note that this in itself is a highly nontrivial problem: the main difficulty lies in generating data that is meaningful in practice, in other words, deciding what kind of hypotheses to make about the assignment of trust and distrust relations and the assignment of item ratings, and their correlations as they would occur in a real social network application with a trust network, while making sure that these hypotheses do not bias the outcome of the experiments. This is challenging because our algorithm is also based on hypotheses about the existence and the correlation of trust and ratings in the data, for example, the hypothesis that ratings from trusted people count more than those from others. Another, perhaps easier, way to address the scarcity of test data is to insert our model in a live recommendation framework.

REFERENCES

ADOMAVICIUS, G. AND TUZHILIN, A. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Engin. 17*, 734–749.

BURKE, R. 2002. Hybrid recommender systems: Survey and experiments. *User Model. User-Adapt. Interact. 12*, 331–370.

GINSBERG, M. 1988. Multi-valued logics: A uniform approach to reasoning in artificial intelligence. *Comput. Intell. 4*, 265–316.

GOLBECK, J. AND HENDLER, J. 2006. Filmtrust: Movie recommendations using trust in web-based social networks. In *Proceedings of the 3rd IEEE Consumer Communications and Networking Conference*. 282–286.

GOLBECK, J., PARSIA, B., AND HENDLER, J. 2003. Trust networks on the semantic web. In *Proceedings of the Conference on Cooperative Intelligent Agents*. Lecture Notes in Artificial Intelligence, vol. 2782, Springer, 238–249.

GOLBECK, J. 2005. Computing and applying trust in web-based social networks. Ph.D. dissertation, University of Maryland at College Park, College Park, MD.

GOLBECK, J. 2006. Generating predictive movie ratings from trust in social networks. In *Proceedings of the 4th International Conference on Trust Management*. Lecture Notes in Computer Science, vol. 3986, Springer, 93–104.

GOLBECK, J., ED. 2009. *Computing with Social Trust*. Springer.

GUHA, R., KUMAR, R., RAGHAVAN, P., AND TOMKINS, A. 2004. Propagation of trust and distrust. In *Proceedings of the 13th International Conference on World Wide Web*. 403–412.

HERLOCKER, J., KONSTAN, J., TERVEEN, L., AND RIEDL, J. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst. 22*, 5–53.

HESS, C. AND SCHIEDLER, C. 2008. Trust-based recommendations for documents. *Artif. Intell. Comm. 21*, 145–153.

HOGG, T., WILKINSON, D., SZABO, G., AND BRZOZOWSKI, M. 2008. Multiple relationship types in online communities and social networks. In *Proceedings of the AAAI Spring Symposium on Social Information Processing*. 30–35.

JOSANG, A. 2001. A logic for uncertain probabilities. *Int. J. Uncert. Fuzziness Knowl.-Based Syst. 9*, 279–311.

JOSANG, A. AND LO PRESTI, S. 2004. Analysing the relationship between risk and trust. In *Proceedings of the 2nd International Conference on Trust Management*. Lecture Notes in Computer Science, vol. 2995, Springer, 135–145.

KUNEGIS, J., LOMMATZSCH, A., AND BAUCKHAGE, C. 2009. The slashdot zoo: Mining a social network with negative edges. In *Proceedings of the 18th International Conference on World Wide Web*. 741–750.

MA, H., LYU, M. R., AND KING, I. 2009. Learning to recommend with trust and distrust relation-ships. In *Proceedings of the 3rd ACM Conference on Recommender Systems*. 189–196.

MASSA, P. AND AVESANI, P. 2004. Trust-aware collaborative filtering for recommender systems. In *Proceedings of the International Conference on Cooperative Information Systems*. Lecture Notes in Computer Science, vol. 3290, Springer, 492–508.

MASSA, P. AND AVESANI, A. 2007. Trust-aware recommender systems. In *Proceedings of the 1st ACM Conference on Recommender Systems*. 17–24.

MASSA, P. AND AVESANI, P. 2009. Trust metrics in recommender systems. In *Computing with Social Trust,* J. Golbeck, Ed., 259–285.

MASSA, P., AVESANI, A., AND TIELLA, R. 2005. A trust-enhanced recommender system application: Moleskiing. In *Proceedings of the 20th ACM Symposium on Applied Computing*. 1589–1593.

O'DONOVAN, J. AND SMYTH, B. 2005. Trust in recommender systems. In *Proceedings of the International Conference on Intelligent User Interfaces*. 167–174.

SCHWEIZER, B. AND SKLAR, A. 1961. Associative functions and statistical triangle inequalities. *Publicationes Mathematicae Debrecen 8*, 169–186.

RESNICK, P. AND VARIAN, H. 1997. Recommender systems. *Comm. ACM 40*, 56–58.

RESNICK, P., IACOVOU, N., SUCHAK, M., BERGSTORM, P., AND RIEDL, J. 1994. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*. 175–186.

SCHAFER, B., KONSTAN, J., AND RIEDL, J. 1999. Recommender systems in e-commerce. In *Proceedings of the 1st ACM Conference on Electronic Commerce*. 158–166.

SINHA, R. AND SWEARINGEN, K. 2001. Comparing recommendations made by online systems and friends. In *Proceedings of the DELOS-NSF Workshop on Personalisation and Recommender Systems in Digital Libraries*.

UCHYIGIT, G. AND MA, M. EDS. 2008. *Personalization Techniques and Recommender Systems*. World Scientific Publishing.

VICTOR, P., CORNELIS, C., DE COCK, M., AND PINHEIRO DA SILVA, P. 2009a. Gradual trust and distrust in recommender systems. *Fuzzy Sets Syst. 160*, 1367–1382.

VICTOR, P., CORNELIS, C., DE COCK, M., AND TEREDESAI, A. 2009b. A comparative analysis of trust-enhanced recommenders for controversial items. In *Proceedings of the 3rd International AAAI Conference on Weblogs and Social Media*. 342–345.

VICTOR, P., CORNELIS, C., DE COCK, M., AND HERRERA-VIEDMA, E. 2010. Bilattice-based aggregation operators for gradual trust and distrust. *World Sci. Proc. Series Comput. Engin. Inf. Sci. 4*, 505-510.

VICTOR, P., CORNELIS, C., DE COCK, M., AND TEREDESAI, A. 2011a. Trust- and distrust-based recommendations for controversial reviews. *IEEE Intell. Syst. 26*, 1, 48–55.

VICTOR, P., DE COCK, M., AND CORNELIS, C. 2011b. Trust and recommendations. In *Recommender Systems Handbook,* P. Kantor, F. Ricci, L. Rokach, and B. Shapira, Eds., Springer, 646–675

VICTOR, P., CORNELIS, C., AND DE COCK, M. 2011c. *Trust Networks for Recommender Systems*. Atlantis Computational Intelligence Systems 4, Atlantis Press.

VICTOR, P., CORNELIS, C., DE COCK, M., AND HERRERA-VIEDMA, E. 2011d. Practical aggregation operators for gradual trust and distrust. *Fuzzy Sets Syst. 184*, 1, 126–147.

WILCOXON, F. 1945. Individual comparisons by ranking methods. *Biometrics Bull. 6*, 80–83.

ZIEGLER, C.-N. AND LAUSEN, G. 2005. Propagation models for trust and distrust in social networks. *Inf. Syst. Frontiers 7*, 337–358.

Received September 2010; revised November 2012; accepted December 2012