Classifying Trust/Distrust Relationships in Online Social Networks

Giacomo Bachi [1] Michele Coscia [1] Anna Monreale [1,2] Fosca Giannotti [2]
[1]KDDLab University of Pisa
Largo B. Pontecorvo, 3, 56127 Pisa - Italy
{coscia, annam}@di.unipi.it, paobachi@gmail.com
[2]KDDLab ISTI-CNR
Via G. Moruzzi, 1, 56124 Pisa - Italy
{fosca.giannotti}@isti.cnr.it

***Abstract*—Online social networks are increasingly being used as places where communities gather to exchange information, form opinions, collaborate in response to events. An aspect of this information exchange is how to determine if a source of social information can be trusted or not. Data mining literature addresses this problem. However, if usually employs social balance theories, by looking at small structures in complex networks known as triangles. This has proven effective in some cases, but it under performs in the lack of context information about the relation and in more complex interactive structures. In this paper we address the problem of creating a framework for the trust inference, able to infer the trust/distrust relationships in those relational environments that cannot be described by using the classical social balance theory. We do so by decomposing a trust network in its ego network components and mining on this ego network set the trust relationships, extending a well known graph mining algorithm. We test our framework on three public datasets describing trust relationships in the real world (from the social media Epinions, Slashdot and Wikipedia) and confronting our results with the trust inference state of the art, showing better performances where the social balance theory fails.**

I. Introduction

In the last years, we witnessed the creation and the success of many web platforms hosting user communities, opinions and collaboration. Web services, like Epinions, allow their user to express their judgment about any kind of product. Popular applications, like collaborative filtering, help retail platforms in the task of suggesting to their users the products they may be interested in, because other similar users found them interesting. Examples of collaborative filtering applications can be found in Amazon and in the Internet Movie Database.

The social dimension plays a crucial role in these applications. An information does not have any value per se: it has value only if we find that its source is reliable and/or we grant our trust on it. In the Epinions platform users can flag another user as "trustworthy" or "untrustworthy", creating a complex network with directed positive and negative links. An interesting problem is then how to infer if a new user, with whom there was no interaction before, can be trusted or not.

Data mining researchers have addressed this problem. In [9] the authors represent a trust/distrust network with a directed signed graph, where users u_1 and u_2 are connected by an edge $(u_1, u_2, +)$ if u_1 trusts u_2; and by an edge $(u_2, u_1, -)$ if u_2 distrusts u_1. They use theories of social balance to infer the sign of any given edge. Social balance states that there are simple graph structures that are balanced and the unbalanced structures tend to evolve eventually into them; therefore, given any edge, its sign is likely to be the one that makes balanced the structure where it appears. If both u_1 and u_2 trust u_3, then it is very likely that they also trust each other, as the triangle with all positive edges is balanced while a triangle with one negative and two positive edges is unbalanced.

Our idea is to use generic subgraphs, discovered from the network, instead of triangles, because larger structures capture more behaviors, going beyond what a triangle can describe.

The authors of [9] do not explore structures more complex than a triangle (i.e. at most three nodes and six directed edges) as there is no social balance theory for structures with four or more nodes. Their results, and the ones presented in [14] with a more descriptive approach, show that these simple structures are very effective in some real world complex networks. Social balance theory works well when we have a classical "I trust you; you trust me" situation when expressing opinions. However, it generally performs better when classifying edges for which we have a lot of contextual information, i.e. where the two endpoints share many common neighbors. It underperforms in more complex relationships, as shown in the network of Wikipedia votes, where a positive link means that a user is supporting the promotion to moderator of another user, and a negative link represents an opposition. In this case, being opposed to the same user does not necessarily entail that we will support each other, as social balance suggests.

In this paper we address the problem of creating a framework for the trust inference, able to infer the trust/distrust relationships in those relational environments that cannot be described by simply using the classical social balance theories of triangles. We show that slightly extending the notion of social balance to groups of four or five nodes can greatly increase the quality of the sign classification.

To do so, we extend a well know algorithm in the graph mining literature [16] to deal with directed signed networks. Since we are interested in what happens only in the direct surroundings of each edge, we decompose the trust network in its ego network components. We mine the trust relationships on this ego network transactional dataset. From the patterns extracted, we create a collection of trust rules, by connecting two different patterns with the same edges but one, forming an association rule. The rules with higher support and confidence are then used as the model for the edge classification.

We test our framework on three public datasets describing trust relationships in the real world. Following [9] we extracted trust networks from the social media Epinions, Slashdot and Wikipedia. We confront our results with the trust inference state of the art. Our results suggest that the state-of-the-art

prediction is currently the best technique for the simplest relationship types (Epinions and Slashdot), while the most complex relational environment (Wikipedia) shows a clear outperform of our method over the state-of-the-art.

Our contribution can be then summarized as follows. Firstly, we extend the social balance analysis with empirical evidences, to include structures that are more complex than simple triangles. Secondly, we create a new trust inference framework, by extending a known graph mining algorithm as a further contribution, whose performance are better than the current trust inference state of the art.

II. Related work

Our paper is based on signed networks, social balance theory edge classification and rule-based link prediction.

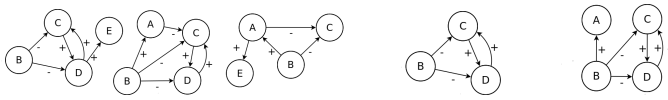
Signed networks are networks where there are two different kinds of edges: positive and negative. Works dealing with signed networks are [9], [14]. In [14] the authors studied the difference between negative and positive interactions between users and which characteristics have the networks with positive or negative connotation, whereas in [15] authors address the study of the existence of different types of trust and how they can be used to improve the performance of some tasks.

Many works study the problem of classifying the edge sign in a social network [9], [6], [11], [4]. The baseline comparison for our framework is [9]. In [9], Leskovec et al. study how it is possible to use various topological features of a social network to classify trust and distrust relations among users in social networks. The authors use the social balance theory (evolving the intuition in [5]) and create 16 rules to classify the sign of an edge, i.e. all the possible subgraphs of three nodes and from three to six edges, each one with two possible signs and directions. They estimate the likelihood of each of these triangles and use the maximum to classify the edge. The methodology works poorly for nodes without a high embeddedness (i.e. for edges whose endpoints do not share many common neighbors), for the subset of edges with high embeddedness the correct prediction rate is remarkably high. Guha et al. [6] present a framework of trust propagation schemes, showing the predicting power of a small number of expressed trusts/distrusts.

The edge sign prediction problem is related to link prediction [10] and link classification [12]. In [1] an approach based on extracting graph evolution rules is described. The model of evolution is learned from the data, by the extraction of evolution rules, used to predict the evolution of the network. [1] it is allowed to predict also when the new links will form. Also [13] addresses the link prediction problem in evolving and multirelational networks, that can be used to represent signed relationships. In [7], authors also analyze the frequent patterns extracted from the network to capture users' behaviors and use them to conduct link formation analysis in directed, temporal social networks. The two works do not consider the sign of the edge.

III. Edge Sign Prediction Problem

In this section, we introduce some preliminary notions, then we present the definition of the edge sign prediction problem; lastly, we describe our approach.



(a) $\mathcal{G}_1$ $\mathcal{G}_2$ $\mathcal{G}_3$ (b) (c)

Figure 1. Support value of two graphs: $supp(b) = 2,\ supp(c) = 1$.

We model a social network by a directed labeled graph denoted by the triple $G = (V, E, \mathcal{L})$, where V is the set of nodes, $E \subseteq V \times V$ is the set of edges and $\mathcal{L} : E \to L$ is a function that assigns labels from the set L to edges. In our setting, we assume that L only contains values $+1$ or -1, i.e., $L = \{+1, -1\}$. Clearly, $\mathcal{L}(u, v) = 1$ denotes that the sign of the edge (u, v) is positive, in contrast $\mathcal{L}(u, v) = -1$ denotes that the sign of the edge (u, v) is negative. We use the triple $(u, v, +1)$ and $(u, v, -1)$ to indicate an edge from a node u to a node v with sign *positive* and *negative*, respectively.

In the following we recall the definition of *subgraph*.

Definition 1: Let $G = (V, E, \mathcal{L})$ and $G' = (V', E', \mathcal{L})$ be two graphs. We say that G' is a *subgraph* of G (or G *contains* G') iff: (1) $V' \subseteq V$, (2) $E' \subseteq E$ and (3) $E' \subseteq V' \times V'$. □

The method proposed in this paper is based on frequent subgraph mining. A problem in this context is how to define the support of a subgraph in order to understand if it is frequent or not. In literature, two approaches have been proposed: one considers the graph G as a single graph and counts how many times G *contains* a graph G' ([2], [8]); the other considers the graph G partitioned in different graphs and to compute the frequency of a graph G' the algorithm counts the number of the graphs that contain G' at least once [16]. We use this last definition and we denote by $supp(G')$ the support of the graph G', i.e., the number of graphs containing G'.

Let us assume $\mathcal{G}$ is a dataset of n graphs. The function $\sigma(G, \mathcal{G})$ is a function that cycles over all graphs $\mathcal{G}_i \in \mathcal{G}$ and checks if G is a subgraph of $\mathcal{G}_i$, according to the Definition 1. If so, it puts $\mathcal{G}_i$ in the set of results and then returns this set ($\sigma(G, \mathcal{G}) = \{\mathcal{G}_i \mid G$ is a *subgraph* of $\mathcal{G}_i \wedge \mathcal{G}_i \in \mathcal{G}\}$). Thus, $supp(G) = |\sigma(G, \mathcal{G})|$. As an example, consider the set of graphs $\mathcal{G} = \{\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3\}$ in Figure 1. We can observe that the support of the graph G_1 (Figure 1(b)) is 2 because it is subgraph of $\mathcal{G}_1$ and $\mathcal{G}_2$, in contrast the support of G_2 (Figure 1(c)) is equal to 1 because it is subgraph of $\mathcal{G}_2$.

We now tackle the problem of predicting the sign of an edge in online social networks. In general, given a network where the sign of a specific edge is hidden/unknown we want to predict the sign of this single edge. The method we propose constructs a prediction model on the known network by using a graph mining approach and then we use this model to predict the missing sign. As in [9] we use the edge with missing sing as test and all the known network as training set.

Definition 2 (Edge Sign Prediction Problem): Let $G = (V, E, \mathcal{L})$ be a network and $E' \subseteq E$ be a set of edges of G, with no sign. If we have information about the sign of all the remaining edges in G, the *Edge Sign Prediction Problem* consists in inferring the sign of edges $(u, v, ?) \in E'$. □

IV. SOLUTION PROPOSED

Now, we present our approach to the problem in Definition 2. In particular, first we describe the computation of our prediction model (Section IV-A) and then we explain how it can be used for the prediction (Section IV-B).

A. Prediction Model

We saw that the analysis of relations between users in social networks is often addressed using solutions based on triangles, mainly because: (1) they are easy to calculate, (2) they are a frequent pattern in social networks and (3) different theories about "social balance" are based on them. Unfortunately, as it has been shown in the experimental section of [9], without contextual information about the edges triangles are not able to capture complex interactions between users. Starting from this observation our idea is to use generic subgraphs, discovered from the network, instead of simple triangles. Our intuition is that more complex structures may capture more complex behaviors, often going beyond what a triangle can describe. Our approach (Algorithm 1) reflects this intuition and uses graph mining techniques to extract the frequent subgraphs, the base of our prediction model, from a network.

Algorithm 1 MineRules(*G,minSup,minConf*)

Require: Directed and weighted graph G and support and confidence thresholds $minSup, minConf$.

Ensure: A set of sign prediction rules R_{set}.

1: DatEgo=**ExtractEgo(G);**
2: FreqPat=**ExtractFreqPat(**$DatEgo, minSup$**)**
3: R_{set}=**CreateRule(**$FreqPat, minConf$**)**

The main steps to construct our prediction model are: generate a ego-network dataset from G; extract frequent subgraphs from the dataset; generate sign rules using frequent subgraphs.

1) Ego-network Extraction: The first step is to generate a subdivision of the network G in different subgraphs, called *ego-networks*. An ego network is a sub-network centered on a particular node who is the subject of the network. The focal point of the network is called the *ego*. In an ego-network, only nodes that are directly connected to the *ego* form the extracted substructure. An ego-network is defined as follows:

Definition 3 (Ego network of a node): Let $G = (V, E, \mathcal{L})$ and v be a network and a node of G, respectively. The *ego-network* of the node v is the sub-network $EG(V', E', \mathcal{L})$, where the set V' is composed of all the neighbors of v (including v itself) and the set $E' \subseteq E$ is the set of all edges in E that are established only between nodes in V'. □

An ego-network enables a focused view on the specific properties of a node highlighting all its interactions with the neighbors. Figure 2 depicts an ego-network example, showing the relations of the node C (the ego) and its neighbors.

In our approach, for each node of the initial network we extract its ego-network. In this way we transform the initial network into a dataset of ego-networks $\mathcal{G}$ (*DatEgo* in Algorithm 1) which is logically equivalent to a transactional dataset, where each transaction is represented by an ego-network related to a specific user (node).

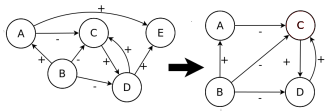
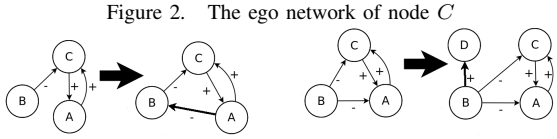


Figure 2. The ego network of node C



(a) (b)

Figure 3. Example of rules: in bold the edge of the prediction

An objection to this approach may be that we are manipulating the original structure, decomposing thus altering the possible results. We defend our choice with two arguments. First, it has already been shown in literature that this approach is able to unveil interesting properties in the case of community discovery [3]. Second, our purpose is to infer a specific property of an edge by analyzing the behavior of the different nodes in the surroundings of the edge itself. We then focus on each node directly affected by the edge itself, as what happens three or four degrees of separation away is not relevant. The next step is to understand how much individuals have the same behavior and so, which behaviors are common.

2) Extraction of frequent subgraphs: At this step, the objective is to extract all the common structures from the ego-network dataset $\mathcal{G}$, obtained in the previous step. We use graph mining techniques for the extraction of frequent subgraphs. We use the notion of frequency of a graph defined in Section III as the number of ego-networks containing the graph ($\sigma(G, \mathcal{G})$). The function *ExtractFreqPat* in Algorithm 1, given the *minSup* threshold, returns the subgraphs contained in at least *minSup* ego-networks ($|\sigma(G, \mathcal{G})| \geq minSup$).

3) Construction of rules: The goal of this step is the generation of rules that are the basis of our approach. Intuitively, a rule models how the users establish relations between them.

Definition 4 (Rule): Let $G_1 = (V_1, E_1, \mathcal{L})$ and $G_2 = (V_2, E_2, \mathcal{L})$ be two graphs. This pair of graphs form a *rule* $R : (G_1 \rightarrow G_2)$ (with G_1 **body** and G_2 **head** of the rule) iff $\exists!(u, v) \in E_2$ s.t. $(u, v) \notin E_1$ and $u \in V_1$. The edge (u, v) is called edge of the prediction. The set of nodes and edges of a rule are respectively $V_R = V_{head}$ and $E_R = E_{head}$. □

This definition allows two types of rules as shown in Figure 3: a rule where the edge of prediction is between two nodes of the body (Figure 3(a)) and a rule where the edge of prediction includes a new node that there was not in the body (Figure 3(b)). Given a rule we can define its *confidence* as follows.

Definition 5 (Confidence of a rule): Let R be a rule of the form $(G_1 \rightarrow G_2)$, and let $supp(G_1), supp(G_2)$ be the support of body and head, we define the *confidence* of a rule R as:

$$Conf(R) = \frac{supp(G_2)}{supp(G_1)}$$

□

The set of rules composing our prediction model is constructed on the frequent subgraphs results:

1) Given the set of extracted frequent subgraphs $\mathcal{F} = \{G_1, G_2, \ldots, G_n\}$, we group them in sets composed of graphs with the same number of edges and we sort the list of these sets in ascending order of number of

edges. So, we obtain $\mathcal{F}_{grouped} = \{\mathcal{G}r_1, \mathcal{G}r_2, \ldots, \mathcal{G}r_m\}$ ($m \leq n$), where the graphs in $\mathcal{G}r_i$ have a number of edges less than the graphs in $\mathcal{G}r_j$ if $i < j$;

2) Starting from the set of graphs with the lowest number of edges $\mathcal{G}r_1$, we take each element of this set as *body* of a rule and for each body we search the graphs that can be *head* of the rule. These graphs are taken from the set of graphs which have exactly one more edge; we call this set G_h. In particular, given a body B we select from G_h all the graphs which can be head of a rule with body B, then we compute for each of them the confidence and we put in our model only the rules that have the confidence greater than a given threshold;
3) After we have taken all graphs of the set $\mathcal{G}r_1$ as body, we repeat the process described in point 2 by considering the elements of the next set of $\mathcal{F}_{grouped}$.

In this way all pairs of subgraphs satisfying Definition 4 and with a confidence (Definition 5) greater than a specific threshold, are included in the rule set, that is our model.

B. Prediction

Once we have constructed the model, i.e. our set of rules, we can use it to predict the sign of a set of edges in a network.

To classify the sign of an edge we have to look for all rules that can be compatible with the edge we want to classify. These rules are called *candidate rules* and are defined as follows:

Definition 6 (Candidate rule): Let $(u, v, ?)$ be an edge in a network G without the sign information, the rule $R = (head \rightarrow body)$, with arch of prediction (u_p, v_p, l_p) is called *candidate rule* for the edge $(u, v, ?)$ if exists a bijective function $f : V_G \rightarrow V_R$ such that:

1) $\forall (k, h) \in G$, $(f(k), f(h)) \in E_R$
2) $\forall (k, h) \in G$, $\mathcal{L}((k, h)) = \mathcal{L}((f(k), f(h))$
3) $(u_p, v_p) = (f(u), f(v))$

□

Thus, after selecting all the candidate rules, we have a set of rules, and each of them provides a sign for the edge to be predicted. If these rules have the same sign, then it is indifferent the rule we choose for predicting the sign; but, if there are rules with different sign we must select the best rule for maximizing the precision of the model.

The best rule among candidate ones for predicting the sign of edge, is the rule which has the highest probability of coming true, i.e., the rule with the best confidence. The confidence of a rule is calculated according to the Definition 5 and is a measure that indicates how often a body evolves in a head.

Algorithm 2 describes the process described above.

Algorithm 2 Predictor(G,$(u,v,?)$,R_{set})

Require: A directed and weighted graph $G(V, E, \mathcal{L})$, the edge $(u, v, ?) \in E$ with omitted sign and the model R_{set}
Ensure: The predicted sign

```
PredictionRule = ∅
for all R ∈ R_set do
    if IsCandidateRule(R) then
        if R.conf > PredictionRule.conf then
            PredictionRule = R
        end if
    end if
end for
return sign(PredictionRule)
```

V. Experiments

This section presents the results of the prediction using the model constructed as described in Sections III-IV on three large online social networks: Epinions, Slashdot and Wikipedia. We describe their characteristics and then show and discuss the classification performances. The aim of the framework is to obtain the rules that better describe the trust structures in the dataset.

	$\|V\|$	$\|E\|$	$\|E_+\|$	$C(u,v) \geq 10$	$C(u,v) \geq 25$
Epinions	119,217	841,200	85%	359,381	211,819
Slashdot	82,144	549,202	77.4%	51,361	22,630
Wikipedia	7,118	103,747	78.7%	61,321	29,610

Table I
Statistics of our networks

Dataset

We tested our framework on the same datasets used in [9], enabling a direct comparison of advantages and weaknesses of an approach based on frequent subgraph mining. In **Epinions** the nodes are the users and an edge is the evaluation made by an user of the opinions of another user, that may be positive or negative. **Slashdot** a news site regarding the world of technology. In the "Slashdot Zoo" a user can mark another user as friend or foe. **Wikipedia** the network is extracted from the votes cast by the users in the elections for promoting users to the role of administrator. A positive/negative edges between users mean votes for/against the promotion.

In Table I are summarized some characteristics of the datasets and we can observe that the sign of edges in network is largely positive (from 77% to 85% of edges have a positive sign). We also report the number of edges with a given minimum embeddedness $C(u, v)$, that counts the common neighbors of the endpoints of the edge, or:

Definition 7 (Embeddedness): Let $G(V, E)$ be a graph. The embeddedness $C(u, v)$ of an edge $(u, v) \in E$ is $C(u, v) = |\{k \in V, k \in Neighbours(u) \wedge k \in Neighbours(v)\}|$, where $Neighbours(u) = \{v \in V, \exists (u, v) \in E\}$. □

The embeddedness is computed not on directed edges, i.e. $Neighbours(u)$ counts the in- and the out-neighbors of u. $C(u, v)$ is used in [9] because the social balance approach performs better for edges with an high embeddedness. However, edges with $C(u, v) \geq 25$ are a small fraction of the network, from slightly more than 4% (Slashdot) to almost 29% (Wikipedia). Good performing rules on these edges do not describe the entire network.

The final piece needed to validate our result is the accuracy function: we follow [9] for comparison purposes. The accuracy function $A(G, M, C_{min})$, given a signed graph G, a model M and a minimum embeddedness C_{min} is defined as:

$$A(G, M, C_{min}) = \frac{|\{(u, v) \in G : Predictor(G, (u, v), M) = sign(u, v)\}|}{|\{(u, v) \in G : C(u, v) \geq C_{min}\}|}$$

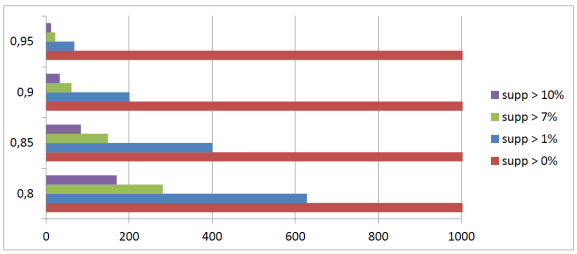


Figure 4. Number of rules from Wikipedia varying $minSup$ and $minConf$.

Results

For a comparison with the performance in [9], we must have the same set up for the experiments. Therefore we created, for each network, a balanced version of the network where the number of positive and negative edges are equal. In the balanced network we have included all the negative edges, and selected at random the same number of positive edges.

Identification of the best performing set of rules. To create our model we need a threshold for support and confidence, with the aim of not extracting and create useless rules, as the mining phase is also time consuming and effective support and confidence thresholds are able to significantly prune the search space, dramatically lowering the time requirements. On the other hand if the thresholds are too high, we may end up excluding some rules that can improve the classification performances. We report in Figure 4 the number of rules generated with different support and confidence thresholds. We can see that without a support threshold, we end up with more than 1000 rules. On the other hand, a 0.8 confidence threshold generates almost 200 rules even with very strict support thresholds. We decided to prune rules with a fairly high confidence (0.95 or more) as this means that when we can apply a rule, we already know that 95% of the outcomes of that rule in the network are described by it. We are less strict on the support, as we want as many rules as we can but not an unmanageable quantity, fixing it as at least 1%.

To better understand how the values of support and confidence affect the classification performances, the experiments were carried out on random subnetworks having 1% of the nodes of the starting network: we have selected at random 1% of nodes of graph and we have created a subgraph with vertex set composed by these nodes and edge set composed by all edge between them. We sampled the networks to test our framework with different threshold values for support and confidence. The results are summarized in Table II.

In Table II, the absolute values of accuracy are not significant. The set of rules were extracted from the original complete networks, but then they were applied on the sampled networks, which do not have the characteristics of the original networks. Also note that the 0% results are generated by the fact that the threshold was too strict, not generating any rule. However, what is interesting is how the accuracy of the classification changes as a function of different thresholds of confidence and support. In particular, demanding a too high confidence with the same support hurts the quality of the results. On the other hand, a higher support threshold usually increase the accuracy

	Supp	Conf	Accuracy 0	Accuracy 10	Accuracy 25
Wikipedia	1%	0.90	70%	65%	63%
	1%	0.98	68%	63%	63%
	10%	0.90	79%	80%	68%
	10%	0.98	0%	0%	0%
Slashdot	1%	0.90	75%	61%	62%
	1%	0.98	54%	61%	62%
	10%	0.90	16%	22%	26%
	10%	0.98	0%	0%	0%
Epinions	1%	0.90	71%	70%	68%
	1%	0.98	0%	0%	0%
	10%	0.90	94%	97%	90%
	10%	0.98	0%	0%	0%

Table II
DIFFERENT ACCURACY FOR DIFFERENT THRESHOLDS AND $C(u, v)$.

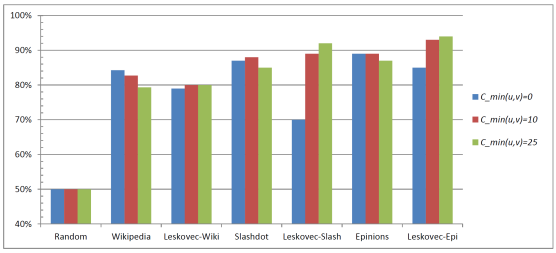


Figure 5. Precision of the prediction of the sign of an edge. Leskovec-* columns are the current state of the art. Color image.

performances (up to the very interesting 97% accuracy for Epinions with 10% support an 0.9 confidence for the edges with at least $C_{min} = 10$). The main explanation is that the high value of confidence threshold excludes many rules, therefore many edges have not a candidate rule and thus they are not classified. We leave as future work the validation of these thresholds not on samples, to validate if the high support, fairly high confidence and a reasonable amount of rules combination can lead to better accuracy also on the complete networks.

Accuracy of prediction. In Figure 5 we show the performances according to the accuracy function with our approach and the classifier based on triangles presented in [9]. The results are considered across different type of edges, according to the $C_{min}(u, v)$ value (column color).

Our observations are the following. The most important information is that our method outperforms the social balance-based triangles method on all networks when considering the total set of edges (the ones with minimum embeddedness equal to 0, blue column). When classifying the sign of all edges of the network, we have a little gain in Epinions, a gain from below 80% to above 80% in Wikipedia, and a jump in Slashdot from 70% to almost 90%. This is due to two factors: we have rules that do not involve only triangles, but more complex structures with four nodes, and we can also ignore misleading triangles, being with a support or confidence not high enough.

A second observation is that the method presented in [9] works best with edges presenting high values of minimum embeddedness. On the other hand our approach based on rules works the other way around (i.e. worse accuracy with high embeddedness): this could be because if an edge is involved in many triangles (high value of embeddedness means many common neighbors, therefore many triangles that contains the edge) it will have many candidate rules with high confidence. Therefore it is more difficult to identify the best rule, resulting in performance degradation. However, we also recall that the set of edges with high minimum embeddedness is small (from

4% to 29% of a network) as reported in Table I. This means that a classifier, to maximize its performances, will use the rules extracted with our model, unless for edges with high embeddedness, for which the triangle approach is preferred.

A third observation concerns our highr prediction quality on the Wikipedia network, that is significantly different and shows different connection dynamics. Regardless of the embeddedness of an edge, in the more complex relation environments the trust relationships can be better modeled by more complex structures extracted with our graph mining approach than the simple ones extracted with the triangle approach.

The particularity of the distrust and of the different trust dynamics. The negative relationships are different in nature than the positive [14], and have a different propagation mechanism on the social networks [6]: $> 60\%$ of the wrong predictions occur when assigning the negative sign. This finding is also emphasized by the most frequent rules used, shown in Figure 6 for the Slashdot (a-d), Epinions (e-h) and Wikipedia (i-l) respectively. These Figures show the rules used more frequently by our framework to predict the sign of an edge. The rules predicting positive signs are more commonly used, again with the exception of the complex Wikipedia case.

Moreover, these rules can be used to have a qualitative description of the trust dynamics in different environments. First, in [9] authors report that the models extracted in one network do not show significant decay in the classification performances when applied to a different network, i.e. the extracted trust dynamics are almost "universal" and captures how trust spread in general, not in particular environments, which are all the same. Our results are very different: there is only one rule shared by two datasets (rules f and l in Figure 6 are isomorphic). There are differences in how the trust propagates in different networks, thus caution in assuming that a model found in a dataset can be applied in a different one.

Rules b, e and i are very similar (though not isomorphic). These rules state that a balanced general consensus (a triangle of positive edges) frequently generates even more consensus, attracting positive edges from nodes that are not part of the triangle. Slashdot users go beyond the one degree of separation of the trust: Figure 6a reports that node C ends up trusting node A even if A is two degrees of trust away from it, thus part of no triangle. In Epinions, negative consensus of another group attracts frequently even more negative opinions, as depicted in Figure 6g. In Wikipedia one would expect that a negatively judged user will cast a positive vote on a user that is negatively judged by the friends of its judges, but Figure 6k goes exactly in the opposite direction. This is another proof of the more complex dynamics present in a voting network.

VI. Conclusion

In this paper we have addressed the problem of creating a framework for the trust inference in a online social network. We proposed a prediction model that is constructed on the known network by using an approach based on graph mining techniques and then we use this model for predicting the trust sign of a given set of edges of the network. Our model

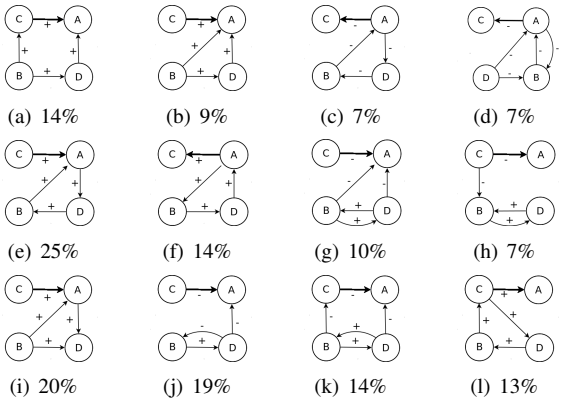


Figure 6. Rules used more frequently to predict the sign of an edge with the percentage of the edges predicted on Slashdot, Epinions and Wikipedia

is able to capture complex relations among users thanks to the use of generic frequent subgraphs discovered from the network, instead of simple triangles. So, it allows us to infer the trust/distrust relationships in those relational environments that cannot be described by simply using the classical social balance theories of triangles. Our experiments have proven that extending the notion of social balance to groups of four or five nodes can increase the quality of the sign classification. On this basis, as a future development, we can create an extended social balance theory, and a novel classification framework based on it, using both graph mining and triangles.

References

[1] M. Berlingerio, F. Bonchi, B. Bringmann, and A. Gionis. Mining graph evolution rules. In *ECML/PKDD (1)*, pages 115–130, 2009.

[2] Björn Bringmann and Siegfried Nijssen. What is frequent in a single graph? In *PAKDD*, pages 858–863, 2008.

[3] Michele Coscia, Fosca Giannotti, and Dino Pedreschi. Towards democratic group detection in complex networks. In *SBP*, pages 105–113, 2012.

[4] T. DuBois, J. Golbeck, and A. Srinivasan. Predicting trust and distrust in social networks. In *SocialCom*, pages 418–424, 2011.

[5] David A. Easley and Jon M. Kleinberg. *Networks, Crowds, and Markets*. Cambridge University Press, 2010.

[6] R. V. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *WWW*, pages 403–412, 2004.

[7] C. W k. Leung, E-P. Lim, D. Lo, and J. Weng. Mining interesting link formation rules in social networks. In *CIKM*, pages 209–218, 2010.

[8] Michihiro Kuramochi and George Karypis. Finding frequent patterns in a large sparse graph*. *Data Min. Knowl. Discov.*, 11(3):243–271, 2005.

[9] J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg. Predicting positive and negative links in online social networks. *WWW*, 2010.

[10] David Liben-Nowell and Jon M. Kleinberg. The link prediction problem for social networks. In *CIKM*, pages 556–559, 2003.

[11] H. Liu, E-P. Lim, H. W. Lauw, M-T. Le, A. Sun, J. Srivastava, and Y. A. Kim. Predicting trusts among users of online communities: an epinions case study. In *ACM-EC*, pages 310–319, 2008.

[12] E. Raad, R. Chbeir, and A. Dipanda. Discovering relationship types between users using profiles and shared photos in a social network. *Multimedia Tools and Applications*, pages 1–30, 2011.

[13] Y. Sun, J. Han, C. C. Aggarwal, and N. V. Chawla. When will it happen?: relationship prediction in heterogeneous information networks. In *WSDM*, pages 663–672, 2012.

[14] M. Szell, R. Lambiotte, and S. Thurner. Multirelational organization of large-scale social networks. *PNAS*, 107(31):13636–13641, 2010.

[15] Jiliang Tang, Huiji Gao, and Huan Liu. mtrust: discerning multi-faceted trust in a connected world. In *WSDM*, pages 93–102, 2012.

[16] Xifeng Yan and Jiawei Han. gspan: Graph-based substructure pattern mining. In *ICDM*, pages 721–724, 2002.