Trust Inference in Online Social Networks

Athanasios Papaoikonomou, Magdalini Kardara, Theodora Varvarigou
{tpap, nkardara} @mail.ntua.gr, dora@telecom.ntua.gr
National Technical University of Athens, 9 Iroon Polytechniou Str., 15773, Zografou, Greece

***Abstract*— We study the problem of trust inference in signed social networks, in which, in addition to rating items, users can also indicate their disposition towards each other through directional signed links. We explore the problem in a semi-supervised setting, where given a small fraction of signed edges we classify the remaining edges by leveraging contextual information (i.e. the users' ratings). In order to model user behavior, we use deep learning algorithms i.e. a variation of Restricted Boltzmann machine and Autoencoders for user encoding and edge classification respectively. We evaluate our approach on a large-scale real-world dataset and show that it outperforms state-of-the art methods.**

Keywords— signed social networks; edge classification; trust; restricted boltzmann machines; autoencoders

I. INTRODUCTION

Social networks have transformed multiple aspects of human behavior and interactions, with their effect on individuals going beyond their online habits. A large spectrum of people's behavior, ranging from leisure pursuits and lifestyle choices to political stance, is largely affected by that of their peers. The internet industry could not remain intact, as customers increasingly base their decisions regarding future purchases on reviews and comments expressed online by their social connections. Online retailers, for whom the ability to predict a user's interests and make accurate recommendations for future purchases is extremely valuable, have quickly realized the role of peer influence in consumer habits and are looking for ways to incorporate social information in their predictive models. Such mechanisms help recommender systems directly address challenges such as the cold-start problem, i.e. how to build an accurate model of a user's profile before they have provided sufficient indications of their preferences through their purchase or ratings history. The role of trust and distrust emerges naturally. In most social networks, connections imply an acquaintance relationship which can range from awareness of each other's existence to close friendship, but show little indication of the levels of trust and distrust between the users. There are, however, web sites, which allow their users to explicitly annotate their disposition towards others as either positive or negative, indicating friendship or enmity respectively. For example in Slashdot, a technology news web site, users are able to indicate their preferences, by tagging each other as friend or foe.

In social networks with no explicitly signed links, inferring trust/distrust is a challenging task, mostly due to the non-transitive nature of distrust, and has been the focus of several previous works. A first attempt to study the connectivity patterns of friendship/enmity networks dates back to the 1940's with the introduction of the structural balance theory by Heider [1], which was later expressed in graph-theoretic language by Cartwright and Harary [2]. The balance theory is applied on simple, undirected networks and studies the distribution of positive and negative edges on triads of users. Even though it was formed under very different conditions, the structural balance theory approximates well the behavior of current real-world signed social networks. An alternate theory of status that was first introduced by Guha et al. in [3] and later established by Leskovec et al. in [4], has recently emerged. According to this new theory, labeled links are not just signs of friendship or enmity but rather impose a hierarchy that can be explained by users' statuses. A positive (/negative) edge from user u to user v means that u regards v as having higher (/lower) status than her. Therefore, the edge sign prediction problem reduces to the estimation of proper status settings for the social network users. In this work, we consider the problem of sign prediction in a semi-supervised setting. By exploiting the users' preferences for certain items, we attempt to reconstruct the signed social graph that depicts the trust/distrust relationships among the users using minimal supervised data. By modeling user behavior accurately, we are able to capture correlations that are useful to the edge sign prediction task. We use deep learning algorithms (i.e. Restricted Boltzmann machine and Autoencoders) for user modeling and evaluate our approach on a large-scale real-world dataset. The evaluation results show that our method outperforms state-of-the art methods and could be of value to several application domains including recommender systems. The rest of the paper is organized as follows. In Section 2 we discuss related work and Section 3 states the problem definition. In Section 4, we present our approach and in Section 5, we evaluate our method through a thorough experimental study on a large-scale dataset from Epinions.

II. RELATED WORK

In this section, we report the state of the art research in trust analysis. First, Guha et al. in [3] develop a framework of trust propagation schemes, based on the exponentiation of the adjacency matrix. In [5] Kunegis et al. study the friend/foe network of the Slashdot Zoo, introducing the signed variants of global, node-level and link-level network characteristics. In [6] the authors view edge sign prediction as a matrix factorization problem while in [7] the authors examine the relative strength between positive and negative edges. In [8] the authors propose a solution to the edge sign prediction problem which involves the discovery of frequent graph patterns that arise in social graphs. Leskovec et al. in [9] propose a logistic regression model that maps each edge to a high-dimensional feature space that comprises the number and type of triads that are defined by the endpoints of the edge and their common neighbors. Finally, in [10] the authors present a mixed effects framework that simultaneously captures users' behavior, social interactions and the interplay between the two. Their framework associates

latent factors to both users and items and accommodates the principles of balance and status from social psychology. The authors show that it is possible to infer signed social ties based mainly on users' ratings, turning an acquaintance network into a signed trust-distrust graph. We consider the results of the works of Leskovec et al. [9] and Yang et al. [10] as our baseline methods.

III. Problem Definition

In this work, we deal with the problem of trust inference in online social networks. Our goal is to predict the level of trust between two users based on their ratings on a set of items. Our dataset consists of the following: a signed social graph $G(V, E, L)$, where V is the set of vertices corresponding to the users , E is the set of (directed) edges and L is a set of labels that can be assigned to the edges, a set of items M, and a set of ratings $R: V \times M \rightarrow \{1..K\}$, which are assigned by user $u \in V$ to an item $m \in M$. Two users correspond to adjacent nodes in the social graph, if they have engaged in some interaction, i.e. expressed an opinion on each other, with the label $l \in L$ of the edge indicating the type of interaction. The semantics are straightforward: A positive link from a node *u* to another node *v* is viewed as an expression of trust or friendship. Likewise, a negative edge is interpreted as a sign of distrust or antagonism. The main focus of this paper is to propose a solution to the edge sign prediction problem in a semi-supervised setting. More specifically:

Problem definition: Given a *partially* labeled social graph G_{known} in which only a small fraction ε% of the links has a sign label, our goal is to "en-sign" the rest of the graph, i.e. to approximate the function $g: U \times U \rightarrow \{-1, +1\}$, which depicts the disposition (trust/distrust) between users, using the rating distribution expressed by R.

The rationale behind our approach is similar to the concept of transfer learning [11] [12] where the knowledge that we acquire in one domain is used to solve a task in another domain. Throughout this work, we assume that the network structure, i.e. the set of edges *E*, is known to us but we only possess label information for a small fraction of the links.

IV. Approach

In this section, we present our method for the edge sign prediction problem. Our approach is influenced by the recent advances in the field of *deep learning*, a popular domain in machine learning research that utilizes neural networks of many layers to achieve better representations of the input data. In our approach we use two of the building blocks of deep learning architectures, namely the Restricted Boltzmann Machine and the Autoencoder. Our method is decomposed in two stages: First, we exploit the rating information expressed by the users in order to assign to each one of them a binary code indicative of their preferences, by employing a collection of Restricted Boltzmann Machines (RBMs). Building on top of these user codes, we then perform the actual classification task using autoencoder networks. Fig. 1 shows a visual summary of our method, and the next subsections elaborate on these concepts.

A. 1st step - User encoding phase

In this step, our goal is to assign a binary low-dimensional code to each user by exploiting her ratings on certain items. To achieve that, we resort to a Restricted Boltzmann Machine (RBM) network, an energy-based, undirected graphical model. In its simplest form, a RBM consists of binary units which are organized in a set of layers, called the "visible" and the "hidden" layer. Observed data are fed to the visible layer, while the units in the hidden layer capture statistical regularities. In this manner, a RBM approximates the input data distribution and can be considered a generative model. In our setup, the data fed to the visible layer corresponds to the rated items and our aim is to orchestrate the training procedure properly in order to identify useful correlations that will be able to explain the rating behavior of the users. In order to permit the RBM to consume rating data , we chose to replace the binary visible layer with a layer of conditional multinomial units [13], achieving a one to one correspondence between the units in the visible layer and the set of available items: each visible unit will be set to the appropriate "one hot" vector representing the rating of the user for the specific item e.g. [0 0 1 0 0] for rating $r = 3$. Furthermore, given that rating data are usually *sparse* (we expect a single user to rate just a small fraction of the available items), we adopted two of the extensions proposed by Salakhutdinov et al. in [14]. First, we assign a different RBM to each user under the constraint that all RBMs share the same *number* of hidden units and we require that each RBM has active visible units only for the items rated by that user. Second, each RBM has a single training case but all the corresponding connection weights and biases are *tied* i.e if two users have rated the same item, their two RBMs must use the same weights between the visible unit for that item and the hidden units. The expressions for the conditional probabilities for the hidden and visible units of an RBM assigned to a random user are:

$$p(h_j = 1|V) = sigmoid\left(a_j + \sum_{i=1}^{m}\sum_{j=1}^{d} v_i^k W_{ij}^k\right),\ j = 1..d \quad (1)$$

and

$$p(v_i^k = 1|h) = \frac{exp\ (b_i^k + \sum_{j=1}^{d} h_j W_{ij}^k)}{\sum_{l=1}^{K} exp\ (b_i^l + \sum_{j=1}^{d} h_j W_{ij}^l)},\ i = 1..m \quad (2)$$

,where m is the number of items rated by the user, $v_i^k = 1$, if the user rated item i with k and 0 otherwise, W_{ij}^k is the symmetric connection weight between hidden feature j and rating k of item i, b_i^k is the bias of rating item i with rating k, $a_j$is the bias of hidden feature $j$and d is the number of units in the hidden layer. The network parameters are learnt by optimizing the Contrastive Divergence (CD) [15], which involves running a Gibbs sampler for a number of steps alternately using the Equations 1, 2, initialized at the training data. Our encoder network operates as follows: First, we train the RBMs of the users for 30 epochs, using 100 units in the hidden feature layer. After the learning phase has completed, we perform an additional pass over the training data and for each case we compute the conditional probability $p(h_j = 1|V)$ from Equation 1. Based on these probabilities, each RBM will decide whether to 'turn on' ($h_j = 1$) or 'turn off' ($h_j = 0$)

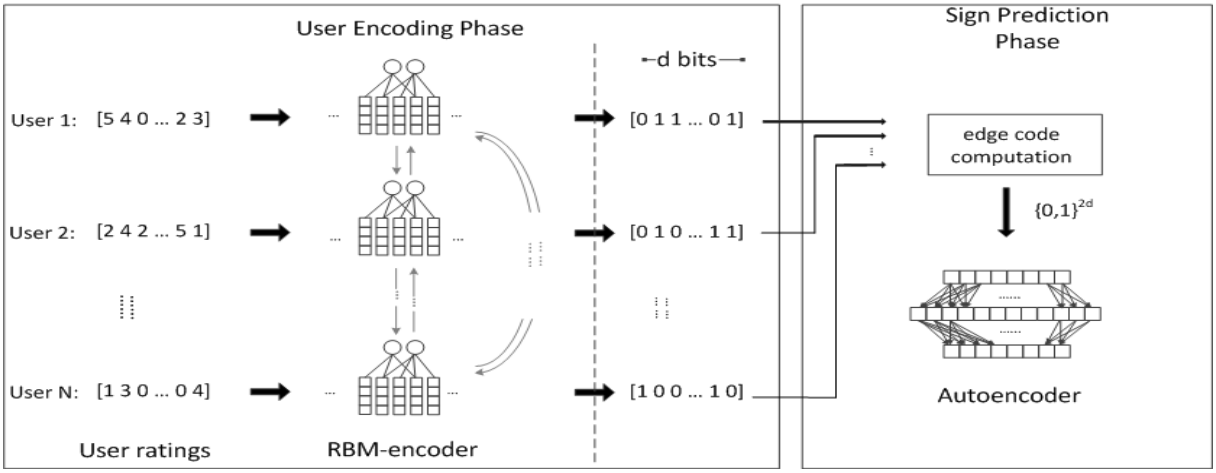

Figure 1 Our approach is decomposed into two steps. a)User encoding phase: We train the RBM encoder using the ratings of the users. A separate RBM is given to each user. After the training is complete, we compute the activations of the hidden units, b) Classification phase : We get the code for each link in the graph, by stacking the user codes of its endpoints. The output is then fed to the autoencoder network to for prediction

$j = 1..d$, the units in its hidden layer. As there is a different RBM for each user, each RBM can set the states of its hidden features independently of the others. Finally, the binary code that each user receives, reflects the states of the hidden units e.g. for states {"off", "off", "on",..} the respective user code will be [0 0 1 ...]. Apparently, the length of the binary code coincides with the number of the hidden units d, which in our case is equal to 100. This step acts as a non-linear dimensional reduction mechanism that allows us to transit from the high-dimensional space of ratings to a much lower dimensional set of binary vectors.

B. 2nd step - Classification with Autoencoder networks

This step performs the actual classification task by employing autoencoder networks which operate on the binary codes assigned to each user from the previous step. An autoencoder neural network is an unsupervised learning algorithm which tries to approximate the identity function by applying non-linear transformations to the input. The architecture of the autoencoder determines the effect that it will exert on the data. For example, autoencoders with fewer hidden units than visible can be regarded as a non-linear dimensionality reduction technique [16]. Other variations entail the *sparse* autoencoder, where a sparsity control is imposed to the neurons of the hidden layer that constrains them to be "inactive" most of the time and the *stacked* autoencoder, which consists of an ensemble of multiple layers of autoencoders, in which the outputs of a layer are fed to inputs of the next layer.

Autoencoders can be particularly useful in the field of semi-supervised learning, where there is an abundance of unlabeled examples and a significantly smaller number of supervised data points. In such cases, an autoencoder is employed to perform unsupervised learning on all the available data. Then, its parameters are used to initialize a supervised neural network that will perform the actual classification task benefiting only from the labeled examples. The underlying hypothesis is that for a supervised problem with domain X and co-domain Y, learning the distribution of $P(X)$ can be useful in approximating the distribution of $P(Y \mid X)$. For the development of our method, we utilized sparse autoencoders with a sparsity target of 0.05, and we experimented with different architectures of stacked autoencoders.

We start by computing a binary code for **each** link in the network, by stacking the codes of its endpoints side by side. Since we deal with a directed network, it is vital to take into account the direction of the edge e.g. for the edge $u \to v$ the respective link code shall be:

$$code(u \to v) := [[\leftarrow code(u) \rightarrow], [\leftarrow code(v) \rightarrow]] \quad (3)$$

Given that each user is encoded with 100 bits (the size of the hidden layer of the RBM encoder) each edge code will be 200 bits long. Next, we feed all the edge codes as input to the autoencoder network and we train it in unsupervised mode. In this way, the autoencoder learns the distribution of edge codes, which we consider crucial for the success of the classification task. Then, after the unsupervised training is complete, we use the parameters to initialize a supervised neural network which will identify *regions* of positive and negative sentiment in the parameter space by exploiting only the labeled examples. The supervised neural network will have an additional layer that will serve as output, encoding the label of the links. We use 'one-hot encoding' for the output i.e. a negative link is encoded as [1 0] and a positive link as [0 1] and standard back-propagation with gradient descent is employed in order to perform the final fine-tuning of the parameters. Finally, we evaluate our method on the remaining edges.

V. EVALUATION

This section presents the results from the evaluation of our algorithm on the social graph of Epinions. We experimented with several autoencoder architectures and we report their predictive accuracy in Table 1. We also report the results of

two baseline methods, namely the work of Yang et al. in [10] and the method of Leskovec et al. in [9]

A. Dataset Description

We have conducted our experiments on a large-scale dataset of 130K users from the product review website of Epinions, collected by Massa et al. [17]. Epinions is an interesting test case because, apart from rating products, it allows its members to directly interact with each other: Users can write reviews for products which are then rated by other individuals and they can explicitly annotate each other as *trusted* or *distrusted*. The dataset in our disposal captures these two interactions: There is a signed social graph formed by 840K trust/distrust statements and a set of 1.5M reviews with 13.6M ratings.

B. Tools and Evaluation measures

The code for the RBM encoder was implemented in the Java programming language and for the analysis with Autoencoders we used the MATLAB deep learning library *DeepLearnToolbox*[1]. The major problem that we faced during the evaluation process was the high skewness of the data, since about 80% of the edges in the signed social graph were positive, allowing any positively biased classifier to achieve remarkable accuracy. To deal with this problem we evaluated our algorithm on a metric that is insensitive to the unequal class distribution, namely the AUC (Area Under Curve) of the ROC (Receiver Operator Characteristic) curve. The experiments were replicated 10 times in order to minimize the effect of randomness.

C. Model Precision and Analysis

Table 1 presents the AUC measurements for different configuration settings of the parameters (autoencoder architectures and fraction of known signed links). To avoid redundancy, the 'network architecture' column refers only to the 'encoder' part of the autoencoder, since the 'decoder' is simply its symmetric image e.g. for the '200-750' architecture of Table 1, there is a '750-200' decoder network that reconstructs the input. Each architecture identifier is in the form $[i-h_1-h_2-..-o]$, where i, o refer to the dimensions of the input and output layers respectively, and the h_i's specify the sizes of the hidden layers. In our case, all the architectures start with a 200-unit input layer in which the edge codes are fed. We have experimented with eight different architectures with 2 to 4 layers and for variable layer sizes. For the first six of them, we set a large first hidden layer ($\geq$500 units) while for the last two we used a 'bottleneck' of 50 units. We trained the architectures with mini-batch gradient descent for 50 epochs and a batch size of 250. The last rows of Table 1 report the measurements of two baseline methods. Some conclusions can be easily drawn:

First, the predictive performance is quite satisfying which supports our choice to model user ratings with deep learning architectures. This procedure provided us with efficient user encoding schemes that were useful for the edge sign prediction task that we examine here. More specifically, the first six architectures demonstrate about 6-10% absolute increase in accuracy for smaller fractions of known links (1% to 30%), and about 2-4% improvement for larger ratios, compared to the baseline [12]. In the cases of the last two (smaller) architectures, our approach works better when the fraction of known signed links is lower than 50% and performs slightly worse for larger percentages. The results indicate that the network complexity introduced by a large hidden layer becomes critical as more labeled data become available. Our method also outperforms the method of Leskovec et al. in [9], in which authors utilize local topology features to train a logistic regression classifier. This method employs a Leave One Out Cross-validation setting, i.e. it predicts the sign of a single edge considering that the edge labels of the rest of the social graph are known, the fraction of known edges thus being close to 100%. In this setting, the authors achieve AUC performance of 0.9342. Our method achieves comparable predictive performance using far less supervised data: We obtain about 0.91 in AUC precision by exploiting only 30% of the available data and we outperform the results in [9] by reaching a success rate of 0.9353 using 90% of the available data and a [200-2000] autoencoder. Second, we deduce that low rank modeling of a signed social graph is indeed possible, since each user can be represented efficiently with a low-dimensional vector. Our approach provides a way to transit from the high-dimensional space of review ratings to a set of much shorter binary codes that are able to capture higher-order correlations in the rating behavior of the Epinions users. To that extent, our results are in line with the work in [18], where sign inference is regarded as a low-rank matrix completion problem. Third, our method can be considered as general-purpose framework, where information can be shared among different machine learning tasks. For example, in the case of a node classification problem, the user codes obtained from the RBM encoder could be directly used as features to the classifier. Finally, we experimented with Support Vector Machines (SVMs) as candidates for the classification task by feeding them with the edge codes of the known links along with their sign. We evaluated both linear and non-linear (mostly Radial Basis Function) kernels and we iterated for a number of rounds using a different random sample from the signed social graph. We found SVMs **not** to be robust when the ratio of labeled examples was small. From iteration to iteration, we noted large fluctuations in predictive performance (from 25% to 70%), which emphasizes the fact that SVMs have a large dependence on the choice of the initial random sample of labeled examples. This is where the ability of deep learning architectures to perform unsupervised learning on the input data pays off. The experiments seem to confirm the *manifold hypothesis* [19] , according to which the generating distribution of real-world high-dimensional data tends to concentrate in the vicinity of a low-dimensional space. The unsupervised training step on the edge codes seems to detect such a low dimensional manifold while the subsequent supervised learning step exploits the labeled data to identify regions of positive and negative sentiment.

[1] https://github.com/rasmusbergpalm/DeepLearnToolbox

Network architecture	Fraction of known labeled links											
	0.1%	0.3%	0.5%	1.0%	5.0%	10%	20%	30%	50%	70%	90%	All -1
200-750	0.7544	0.7687	0.7780	0.8016	0.8476	0.8687	0.8846	0.8947	0.9070	0.9153	0.9232	0.9493
200-1000	0.7690	0.7707	0.7873	0.8060	0.8535	0.8801	0.8988	0.9105	0.9200	0.9284	0.9338	0.9554
200-2000	0.7345	0.7813	0.7984	0.8156	0.8551	0.8780	0.8955	0.9082	0.9195	0.9295	0.9353	0.9592
200-500-200	0.7405	0.7731	0.7776	0.8084	0.8491	0.8623	0.8878	0.8968	0.9115	0.9209	0.927	0.9431
200-1000-500-200	0.7267	0.7802	0.7844	0.8004	0.8401	0.8608	0.8809	0.8936	0.9074	0.9185	0.9280	0.9415
200-800-800-2000	0.7134	0.7955	0.8029	0.8066	0.8443	0.8563	0.8778	0.8938	0.9053	0.9195	0.9269	0.9429
200-50	0.7450	0.7564	0.7762	0.7916	0.8360	0.8580	0.8648	0.8683	0.8753	0.8807	0.8853	0.9070
200-50-50	0.7389	0.7627	0.7847	0.7739	0.8272	0.8345	0.8442	0.8499	0.8624	0.8686	0.8766	0.8904
BRI [10] (Baseline)	-	-	-	0.731	0.747	0.776	0.818	0.836	0.869	0.894	0.912	-
LOO-LR [9] (Baseline)	-	-	-	-	-	-	-	-	-	-	-	0.9342

Table 1. AUC measurements of the experiments for all the different architectures of the stacked autoencoders, and for variable fraction of the known signed links. The network architecture refers to the "encoder" part of the architecture. The last rows present the results of two baseline methods

VI. Conclusions

In this work, we investigated the interplay between user preferences and trust in signed social networks. We studied the problem in a semi-supervised setting and used a small fraction of signed edges to classify the remaining edges by leveraging contextual information (i.e. the users' ratings). We used deep learning algorithms i.e. a variation of Restricted Boltzmann machine and Autoencoders for user encoding and edge classification respectively. We evaluated our method on a large-scale dataset from Epinions. Our findings show that our approach outperforms current state-of-the art methods [10]. In the future, we plan to find ways to incorporate existing social theories in our model, such as the structural balance theory and to examine whether our approach can be generalized to also handle other kinds of edge labeling, besides just positive and negative.

Acknowledgements

This work has been supported by the EU project SUPER (FP7- 606853)

References

[1] F. Heider, "Attitudes and Cognitive Organization," *Journal of Psychology*, vol. 21, pp. 107-112, 1946.

[2] D. Cartwright and F. Harary, "Structural Balance : a generalization of Heider's theory," *Psychological Review*, vol. 63, no. 5, pp. 277-293, 1956.

[3] R. Guha, R. Kumar, P. Raghavan and A. Tomkins, "Propagation of trust and distrust," in *Proceedings of the 13th international conference on World Wide Web (WWW '04)*, New York, NY, USA, 2004.

[4] J. Leskovec, D. P. Huttenlocher and J. M. Kleinberg, "Signed networks in social media," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*, Atlanta, Georgia, USA, 2010.

[5] J. Kunegis, A. Lommatzsch and C. Bauckhage, "The slashdot zoo: mining a social network with negative edges," in *Proceedings of the 18th international conference on World wide web (WWW '09)*, New York, NY, USA, 2009.

[6] P. Agrawal, V. K. Garg and R. Narayanam, "Link Label Prediction in Signed Social Networks," in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence,(IJCAI 2013)*, Beijing, China, 2013.

[7] A. Papaoikonomou, M. Kardara, K. Tserpes and T. A. Varvarigou, "The Strength of Negative Opinions," in *Engineering Applications of Neural Networks*, Xalkidiki, Thessaloniki, Greece, 2013.

[8] A. Papaoikonomou, M. Kardara, K. Tserpes and T. Varvarigou, "Predicting Edge Signs in Social Networks Using Frequent Subgraph Discovery," *IEEE Internet Computing*, vol. 18, pp. 36-43, Sept-Oct 2014.

[9] J. Leskovec, D. Huttenlocher and J. Kleinberg, "Predicting positive and negative links in online social networks," in *Proceedings of the 19th international conference on World wide web (WWW '10)*, New York, NY, USA, 2010.

[10] S.-H. Yang, A. J. Smola, B. Long, H. Zha and Y. Chang, "Friend or Frenemy?: Predicting Signed Ties in Social Networks," in *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '12)*, New York, NY, USA, 2012.

[11] R. Raina, A. Battle, H. Lee, B. Packer and Y. A. Ng, "Self-taught learning: Transfer learning from unlabeled data," in *Proceedings of the Twenty-fourth International Conference on Machine Learning*, Corvallis, Oregon, 2007.

[12] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Trans. on Knowl. and Data Eng.*, October 2010.

[13] G. E. Dahl, R. P. Adams and H. Larochelle, "Training Restricted Boltzmann Machines on Word Observations," in *In Proceedings of the 29th International Conference on Machine Learning*, 2012.

[14] R. Salakhutdinov, A. Mnih and G. Hinton, "Restricted Boltzmann machines for collaborative filtering," in *In Proceedings of the 24th international conference on Machine learning (ICML '07)*, Corvallis, Oregon, 2007.

[15] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771 - 1800, August 2002.

[16] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504 - 507, 28 July 2006.

[17] P. Massa and P. Avesani, "Trust-aware bootstrapping of recommender systems," in *ECAI Workshop on Recommender Systems*, 2006.

[18] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio and P.-A. Manzagol, "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion," *The Journal of Machine Learning Research*, December 2010.

[19] H. Narayanan and S. Mitter, "Sample complexity of testing the manifold hypothesis," in *Advances in Neural Information Processing Systems 23 (NIPS 2010)*, Vancouver, Canada, 2010.