Course Project Title

Social Network Analysis for Computer Scientists — Course Project Paper

Name One
LIACS, Leiden University
sXXXXX@umail.leidenuniv.nl

Name Two
LIACS, Leiden University
sXXXXX@umail.leidenuniv.nl

ABSTRACT
Abstract.

Categories and Subject Descriptors
H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms
Theory

Keywords
ACM proceedings, LaTeX, text tagging

1. INTRODUCTION
One of the main ideas behind link prediction is that it offers several ways in which one can study networks of any kind in which data scientists, software engineers and researchers could greatly benefit from. Social networks are a popular way to model the interactions among people in a group or community. Their connections can be visualized as a graph, where a vertex corresponds to a person in a group and an edge represents some form of association between the corresponding persons.

Organizations, such as Amazon, can extract information based on raw data in order to predict what the customer might actually buy or find interesting. Another example would be the professional social-media platform, Linked-in. Based on incoming data they would be able to, for instance predict your next connection; link back jobs which are relevant to you; link contents/articles based on your connections, interests, and things you read on the web. Similarly, security corporate companies could more precisely focus their efforts based on probable relationships in malicious networks that have heretofore gone unobserved, and researchers can easily adapt link prediction methods to identify links that are surprising given their surrounding network, or links that may not exist at all [15].

That being said, links or associations are usually based on someone's behavior and interests. In social networks however, objects are dynamic given that the number of edges and nodes are changing continuously. Getting a hold of those dynamics [number of nodes and edges] is one of the fundamental problems in social network analysis. One of the essential question(s) which we are going to cover in this paper is, given a network at time t, accurately predict the links that are going to form at a future time t'. This is known as the link prediction problem.

In this paper, we approach the problem of link prediction using supervised learning [9], [10], [15], [23]. Unsupervised techniques have also been used to solve the problem of link prediction. The main difference between the two is that in the unsupervised case, we calculate metrics that individually provide predictions of whether a link will form or not. Supervised learning employs a more sophisticated framework where each of these metrics could be considered as an input to a machine learning model.

This paper is heavily based on methods and techniques described in [10] and [15]. Both papers consider collaboration networks of co-authors on which several machine learning models are applied in order to predict future links (collaborations) between co-authors. Under sampling is used in both papers in order to solve the problem of class imbalance and decrease the size of the training sets. Additionally, in [15], the training set is constructed by considering the n-neighborhood of each node. Moreover, it is proposed to use a specific type of decision tree, called Hellinger Distance Decision Tree (HDDT) [2], [3], applied in the original distribution of the data, that is without employing under sampling. Those trees use the Hellinger distance as splitting criterion, which is skew insensitive [3], and therefore is very well suited for classification with imbalanced data. Although this recommendation is provided, the authors do not present results of this method. The contributions of this paper are as follows:

1. Although there is an in-depth comparison of Hellinger Distance Decision Trees and regular Random Trees (RT) (that use entropy as splitting criterion) in [3], to the best of our knowledge, Hellinger Distance Decision Trees have not been applied to solve the problem

SNACS '17 Social Network Analysis for Computer Scientists, Master CS, Leiden University (`liacs.leidenuniv.nl/~takesfw/SNACS`).

of link prediction, in which the class imbalance could be as high as 1:20000

2. We compare two methods based on [10] and [15]. The first method consists of under sampling the data and fitting RT and HDDT, whereas in the second method we fit those models to the original distribution of our data, that is, without under sampling.

3. Regarding the evaluation of results, apart from the ROC/AUC metric which is proposed in [15], we also use the Precision-Recall curve as suggested in [21], which is better suited for imbalanced data sets.

The paper is structured as follows. In Section 2, we formalize the link prediction problem. Specifically, we discuss the transformation of a network to an ordinary data set, and the class imbalance problem. Related work is provided in Section 3. In Section 4, we describe in detail the methods compared in this paper. Section 5 describes the data sets in which the experiments and results of Section 6 are based on. Finally, we conclude with Section 6.

2. PROBLEM STATEMENT

Consider at a particular time t, an undirected network $G = \langle V, E \rangle$, where V is the set of nodes and E is the set of edges. The goal is to predict the edges that will form in the network at time $t' > t$. As an example, consider a collaboration network of co-authors. Two authors v and w are linked when they have published at least one research paper together. The goal is to predict future connections between co-authors that have not yet collaborated.

In this paper, we approach the problem of link prediction using supervised learning. In this framework, in order to build models with predictive ability, one needs to construct two sets, mainly a *train* and a *test* set. Then, the model learns features and patters from the train set, and it is evaluated on the test set. These two sets form together the *data set*. This set is of the form $(\vec{x}, y)$, where $\vec{x}$ indicates the inputs to the model, and y indicates the label, that is whether or not a link is present between two nodes. The model learns from the instances $(\vec{x}, y)$ of the train set. In the evaluation phase, we feed the instances $\vec{x}$ of the test set to the trained model, and the output is the predicted label, i.e our prediction of whether or not a link will form between two nodes.

We face two problems here. First, how to construct the train and test set, and secondly, what are $\vec{x}$, y in our model.

2.1 Constructing Data Sets

The process of constructing the train and test set is described in detail in [10], [15]. In order to transform the network to a data set, we first choose two adjacent periods, the *train* and *test period*. In the train period, denoted by $[0, t_x]$, we have pairs of authors that are connected, that is, they have written at least one research paper together. The same holds for the test period $[t_{x+1}, t_y]$. We construct a new network $G_x = \langle V_x, E_x \rangle$ consisted of all pair of authors that do not share a connection in the train period. We then examine whether or not they co-authored a paper in the test period. If they did, we assign to the pair a positive label, and otherwise a negative label. The data set contains $\binom{|V_x|}{2} - E_x$ instances, since we do not account for self-loops, and in undirected networks each pair is recorded once.

Selecting an appropriate set of features is one of the most important part in supervised learning algorithms [10]. The inputs to the model are selected by calculating topology-based and node-based metrics from G_x that will serve as attributes to the data set. Such metrics include, but not limited to, *common neighbors*, *Jaccard coefficient*, *Adamic Adar*, *preferential attachment*, *Katz* measure, *PropFlow*, *SimRank*, and many others. All these features can be used individually to provide naive predictions. This is sometimes called the unsupervised framework of link prediction. In the supervised case, we use state-of-the-art predictive models such as Neural Networks, Support Vector Machines, Decision Trees, and Random Forests that treat these individual metrics as inputs. The output or prediction of the model, is whether or not a link will form between two specific nodes.

That being said, we end up with a data set in the standard format $(\vec{x}, y)$, where $\vec{x}$ are metrics extracted from G_x and y indicates the label, that is whether or not a pair of nodes is connected or not.

2.2 Class Imbalance and Size

Although we have constructed the data set, there are two main problems. First, the size of the data set is very large which can make the training process infeasible. For example, consider the relatively small in size `ca-CondMat` network [12], consisted of 23133 nodes and 93497 edges. Defining the train period to be two-thirds of the networks' edges, and assuming that all node appear in V_x, the resulting data set would be of size $\binom{23133}{2} - 62631$, which yields more than 267 million instances. Additionally, the distribution of the classes is highly imbalanced. That is, the amount of instances belonging to the negative class is overwhelmingly greater than the instances of the positive class. Those two issues make the training process difficult or even impossible. There mainly exist two approaches to solve these problems.

One method consists of under sampling the data set to balance in order to create classes of almost the same size. In this way, the size of the training set decreases, and at the same time we obtain balanced classes in which we can fit ordinary predictive models such as, Decision Trees, Support Vector Machines, Neural Networks etc. Note that many of these models perform poorly in highly imbalanced data.

A second method is proposed in [15]. Specifically, it is suggested to treat each neighborhood as a separate problem. We define the n-neighborhood of a node v as the set of nodes that are 0 to n steps away from v. Instead of considering all pairs of nodes to enter the data set, we choose for each node, and for a specified n, only those that belong to the n-neighborhood of the node. In this way, the size of the data set decreases greatly. Furthermore, it is proposed to implement a particular type of decision trees, called Hellinger trees [2], [3], without altering the distribution of the data, that is without under sampling. Hellinger trees use the Hellinger distance as splitting criterion, which is skew insensitive [3].

3. RELATED WORK

The link prediction problem was formalized in [13], where there is an extensive analysis of different baseline predictors, applied to co-authorship networks. Nevertheless, state-of-the-art supervised learning models are not studied in this paper. Furthermore, co-authorship networks have been analyzed in [10], [18], and [5], but this time several machine learning algorithms such as Decision Trees, SVM, Neural Networks, and K-Nearest Neighbors are used for prediction. In contrast to [10], where static, unweighted networks are used, [18] takes into account the evolution of the network, and the authors of [5] analyze a weighted network. Several recommendations and suggestions are presented in [15], regarding how to perform link prediction, and how to approach inherent issues such as dealing with class imbalance and huge train sets.

In the domain of counter-terrorism, link predictions has been studied in [6], [9]. In the latter paper, several networks are constructed by removing nodes from the original graph, called visible networks. A set of features is calculated from these networks in order to predict links of the original one, using supervised learning models. A more probabilistic approach is taken in [7], where chance-constrain programs are used, and in [22], in which the framework of Relational Markov Network is utilized. Finally, [24], and [11] are extensive surveys regarding the different metrics, classification methods, and approaches that have been used for link prediction.

4. SUGGESTED APPROACHES

In this section, we describe the approaches that have been taken in [10] and [15] and we discuss the differences between them. Our approach, which is based on both papers, is applied in 5 social networks with the aim to compare DT which are applied in [10] and HDDT which are proposed in [15], but the authors have not provided results for the latter method.

The authors in [10] use two undirected, unweighted collaboration networks of co-authors, mainly the BIOBASE [1] and DBLP [2] networks. An edge between two authors/nodes exists in case they have written at least on research paper together. Both networks contain timestamps of the years of collaborations. For each network, the authors choose a corresponding train and test period that span one or more years. For example, in the first network, the train period is consisted of 5 years from 1998 to 2002, whereas the test period comprises the year 2003. The data set constructed from this network contains each combination of co-authors that do not exist in the train period, that is, each pair that could potentially have a connection in the future. For each of these combinations, it is examined whether or not there is a connection in the test period. If a connection exists, a positive label is assigned to the pair, otherwise a negative label. Thus, the problem of link prediction has been framed as a binary classification problem in which any known machine learning classification model can be applied in order to distinguish between the positive and negative classes.

[1] `http://www.elsevier.com`
[2] `http://dblp.uni-trier.de/xml/`

For each resulting pair of nodes, several topological and domain specific features are calculated. The former kind of features include the *shortest dictance* and *clustering index*, whereas the domain specific features include the sum of papers the two authors share, the sum of the count of keywords in the papers of each author, the number of common keywords two authors have used etc.

One of the main characteristics of the link prediction problem is the great class imbalance between positive and negative classes. This is due to the fact that the number of potential links that could form is enormously larger than the links that do actually form [15]. Additionally, the resulting data set has a very large size that could make the training process of a classifier even infeasible. To see this, refer to the example described in Section 2.2. In [10] these problems are solved by under sampling the data to balance. Under sampling is a widely used technique in order to cope with imbalanced classes in which we only keep a representative sample of the large class that is more or less equal to the size of the small class. Specific ratios of imbalance can also be specified in the sampling process. The resulting data set is comprised from the sample of the large class and all instances of the small class, resulting into a balanced data set.

After creating a balanced data set, the authors train several machine learning models such as Decision Trees (with and without Bagging), Support Vector Machines, Naive Bayes, Neural Networks and others. They found that Bagging and Support Vector Machines provided the best performances when compared to different metrics such as accuracy, precision, recall, F-value and squared error. The models are evaluated using 5-fold cross validation.

The second paper in which this work is heavily based, is that of Lichtenwalter et al. [15]. The techniques described in this paper are applied in two networks, one weighted, undirected co-authorship network of 19464 edges of condensed matter physics collaborations from 1995 to 2000 called `condmat`, and one directed weighted network of 712 million cellular phone calls, called *phone*. In principle, the construction of train and test sets is the same, that is, topological and node attribute measures are extracted from the network constructed in the train period, and the labeling is being done using the network of the test period.

The authors of the paper treat the problem of class imbalance in great detail. They suggest to construct the data set in the following way. Instead of considering all possible combinations of pairs of nodes that do not exist in the train period, we construct the data set only from those combinations that are not part of the train period, but also contained in the neighborhood of n size. Specifically, for each node v, we calculate its n-size neighborhood. We then compute the metrics between v and the nodes of this neighborhood, by excluding pairs that already belong to the train period. The idea behind this is that, it is more possible for links to form in the future between nodes that are closely together, than with nodes that are far apart. Thus, reducing the problem to each neighborhood offers two advantages. Firstly, the imbalance between classes is decreased, and secondly given that specific metrics such as *Adamic Adar* and *Rooted PageRank*

are expensive to compute, we avoid calculating those metrics for pairs of nodes that are not highly likely to connect in the future. The value of n is chosen by the user. Selecting higher values guarantee that we do not lose connections that actually do form, but on the other hand the instances of the negative class increase dramatically, along with the computation time of the metrics.

Although choosing smaller values for n mitigates the problem of class imbalance, the distributions' skew is still heavy, with the negative class dominating the positive. As an example, for the `condmat` network in [15], values of $n = 2$ and $n = 4$ result in classes of 179:1 and 6247:1, respectively. For the larger `phone` network, the imbalance is even stronger. For the aforementioned values of n, the ratios are 131:1 and 32880:1, respectively.

Apart from the aforementioned suggestion of treating the problem locally for each neighborhood, the authors in [15] also provide their insights and suggestions regarding sampling techniques. One technique that can be used is called SMOTE [1], in which the minority class is over sampled in order to balance the data set. Although that could work in a network such as `condmat`, it will certainly not for `phone`, due to its size [15]. The problem is that by using over sampling techniques we increase the size of the train set which we wanted to reduce in the first place. Regarding under sampling, the authors experiment with different sampling ratios, that is, they explicitly under sample the data to different ratios than 1:1, and then they evaluate a C4.5 Decision Tree [19] by reporting the AUC measure of the respective ROC curve. The idea is that by keeping a ratio of imbalance in the under sampled data set, we retain information from the network, which we might lose when under sampling to 1:1 ratio. Although this is a valid approach, the authors evaluate the classifier trained on imbalanced data using the AUC of the ROC curve, which might be misleading, since the ROC curve is insensitive to skew distribution [21].

Additionally, the High Performance Link Prediction (HPLP) framework is presented in [15]. This framework comprises several topological metrics that can be computed for every network such as *In/Out Degree*, *Common Neighbors*, *Maximum Flow*, *Adamic Adar*, *Jaccard Coefficient*, and others. Note that this is different from the work of Hasan et al. [10] where most of the selected features are domain specific. Hence, HPLP serves as a general framework for link prediction. Additionally, a new unsupervised prediction method is presented called `PropFlow`. It is a predictor based on random walk that starts from node v_i and ends at a node v_j in l or fewer steps, using link weights as transition probabilities [15]. A score s_{ij} is produced for each pair of nodes v_i and v_j that serve as an estimation of the likelihood that the two nodes will be connected in the future. `PropFlow` is also included in HPLP.

Finally, after under sampling to balance both data sets, the authors apply Random Forests with Bagging, using the features proposed in the HPLP framework, and for different values of n. They found that with increasing values of n there is a deterioration in performance, in both networks. The value of $n = 2$ provided the best performance.

In this paper, we compare Decision Trees and Hellinger Distance Decision Trees with bagging as applied and proposed in [10] and [15], respectively, for the case of under sampling to balance (ratio 1:1), and for an imbalanced ratio 1:6, for 5 social networks. To the best of our knowledge, Hellinger Distance Decision Trees have not been applied to solve the link prediction problem. We are interested to examine whether or not HDDT provide sufficiently good results after constructing a data set from a social network and under sampling to a ratio of 1:6.

5. DATA SETS AND SOFTWARE TOOLS

Almost all link prediction works need to verify their methods on the collected datasets. The datasets are important for fairly reproducing and comparing different link prediction methods. Constructing and collecting the datasets is a time-consuming and labor-intensive work. That said, not all the datasets are publicly available to use and some of them are incomplete. Since the process of transforming the network to a data set requires to create two non-overlapping periods that simulate the formation of links between nodes from one period to the other, it is essential to consider networks that contain timestamps. We are interested in both directed and undirected networks. Nevertheless, we convert the former type to the latter. In all of our experiments, we consider unweighted networks.

We use two networks from that come from the KONECT [3] database such as the `UC Irvine messages` [4] [16], a directed network containing sent messages between the users of an on-line community of students from the University of California, Irvine. Next, `Digg` [5][4] is a reply directed network of the social news website Digg where each node in the network is a user of the website, and each directed edge denotes that a user replied to another user. Furthermore, we use the `Dublin Contacts` [6] [20] undirected network in which a node represents a human and an edge corresponds to a contact in the physical world. Moreover, we consider the `IA-Reality-call` [7] [8], [20] an undirected network that consists of human mobile phone call events between a small set of core users at the Massachusetts Institute of Technology (MIT) whom actually were assigned mobile phones for which all calls were collected. The network also contains edges between users that do not belong in the small set of users who called other individuals that were not actively monitored. Therefore, some sort of noise might exist in this network, since we do not have more information about the individuals outside of the small set of MIT users. Finally, we consider the directed `MathOverflow` [8] [17] network, where a node represents a user, and an edge indicates that user u answered user's v question on the website.

Although there are many link prediction metrics and meth-

[3] `http://konect.uni-koblenz.de/networks/`
[4] `http://konect.uni-koblenz.de/networks/opsahl-ucsocial`
[5] `http://konect.uni-koblenz.de/networks/munmun_digg_reply`
[6] `http://networkrepository.com/ia-contacts-dublin.php`
[7] `http://networkrepository.com/ia-reality-call.php`
[8] `http://snap.stanford.edu/data/sx-mathoverflow.html`

ods proposed, only very few works open their source codes. Re-implementing methods and formulas to calculate predictors is a time-consuming process. Only few public tools try to integrate these metrics and methods such as `linkpred`[9] and `LPmade` [10] [14] which both have a handful of link prediction metrics. `LPmade` is a cross-platform software solution that provides multi-core link prediction and related tasks and analysis [14]. It is written in `C++` and therefore it is suited for handling very large networks. Unfortunately, compilation issues and furthermore the lack of documentation and support prevented us from using the software. Hence, we used a `Python` library called `linkpred`. Unfortunately, due to the fact that it is entirely written `Python` it can prove to be very slow for handling large networks.

6. REFERENCES

[1] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[2] D. A. Cieslak and N. V. Chawla. *Learning Decision Trees for Unbalanced Data*, pages 241–256. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[3] D. A. Cieslak, T. R. Hoens, N. V. Chawla, and W. P. Kegelmeyer. Hellinger distance decision trees are robust and skew-insensitive. *Data Mining and Knowledge Discovery*, 24(1):136–158, Jan 2012.

[4] M. De Choudhury, H. Sundaram, A. John, and D. D. Seligmann. Social synchrony: Predicting mimicry of user actions in online social media. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 4, pages 151–158. IEEE, 2009.

[5] H. R. De Sá and R. B. Prudêncio. Supervised link prediction in weighted networks. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 2281–2288. IEEE, 2011.

[6] M. Dombroski, P. Fischbeck, K. Carley, et al. Estimating the shape of covert networks. In *Proceedings of the 8th International Command and Control Research and Technology Symposium*, 2003.

[7] J. Doppa, J. Yu, P. Tadepalli, and L. Getoor. Chance-constrained programs for link prediction. In *NIPS Workshop on Analyzing Networks and Learning with Graphs*, 2009.

[8] N. Eagle and A. Pentland. Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, 2006.

[9] M. Fire, R. Puzis, and Y. Elovici. *Link Prediction in Highly Fractional Data Sets*, pages 283–300. Springer New York, New York, NY, 2013.

[10] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.

[11] M. A. Hasan and M. J. Zaki. *A Survey of Link Prediction in Social Networks*, pages 243–275. Springer US, Boston, MA, 2011.

[12] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data*, 1(1), Mar. 2007.

[13] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, CIKM '03, pages 556–559, New York, NY, USA, 2003. ACM.

[14] R. N. Lichtenwalter and N. V. Chawla. Lpmade: Link prediction made easy. *Journal of Machine Learning Research*, 12(Aug):2489–2492, 2011.

[15] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla. New perspectives and methods in link prediction. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 243–252, New York, NY, USA, 2010. ACM.

[16] T. Opsahl and P. Panzarasa. Clustering in weighted networks. *Social networks*, 31(2):155–163, 2009.

[17] A. Paranjape, A. R. Benson, and J. Leskovec. Motifs in temporal networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, pages 601–610, New York, NY, USA, 2017. ACM.

[18] M. Pavlov and R. Ichise. Finding experts by link prediction in co-authorship networks. In *Proceedings of the 2Nd International Conference on Finding Experts on the Web with Semantics - Volume 290*, FEWS'07, pages 42–55, Aachen, Germany, Germany, 2007. CEUR-WS.org.

[19] J. R. Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[20] R. A. Rossi and N. K. Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[21] T. Saito and M. Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[22] B. Taskar, M. fai Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In S. Thrun, L. K. Saul, and P. B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 659–666. MIT Press, 2004.

[23] C. Wang, V. Satuluri, and S. Parthasarathy. Local probabilistic models for link prediction. In *Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, ICDM '07, pages 322–331, Washington, DC, USA, 2007. IEEE Computer Society.

[24] P. Wang, B. Xu, Y. Wu, and X. Zhou. Link prediction in social networks: the state-of-the-art. *CoRR*, abs/1411.5118, 2014.

[9] `https://github.com/rafguns/linkpred`

[10] `https://github.com/rlichtenwalter/LPmade`