User diversity in social networks and the impact of link recommendation algorithms on it

Styliani Bourli

Diploma Thesis

Supervisor: Prof. Evaggelia Pitoura

Ioannina, July, 2019

ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
UNIVERSITY OF IOANNINA

Acknowledgments

I would like to express my sincere thanks to Mrs Evaggelia Pitoura, for the opportunity she has given me to perform this work in an interesting field such as the analysis of social networks as well as the important help, support and guidance she has provided me.

03/07/2019
Styliani Bourli

Abstract

By social network we usually refer to a set of entities and how they interact with each other. Typically, the social network is represented as a graph in which the entities (individual people, or things) are the nodes and interactions (or relationships) between them are the edges, or links. With regard to social networks, we study the problem of low diversity between linked nodes. Low diversity exists when the nodes tend to be linked to similar others. More over we study the effect of link prediction and link recommendation algorithms on diversity; if they also suggest to nodes, similar ones for connection. By defining similarity, analyzing some social networks and performing some experiments, we prove that the diversity is actually low and algorithms affect it. We also explain why this is a bad phenomenon.
Specifically, this diploma thesis consists of five chapters. The first chapter contains an introduction to the problem we are studying. The second chapter refers to link prediction and recommendation algorithms used for the experiments. The third chapter concerns the creation of a synthetic network, experiments on it and conclusions that come out. Accordingly, the fourth chapter concerns the creation of a network of actual data, experiments on it and the conclusions that come out. Finally, the fifth chapter summarizes the whole process and the conclusions that have emerged and contains some ideas for future work.

Keywords: Social network analysis, link prediction, link recommendation, diversity

Περίληψη (extended abstract in Greek)

Με τον όρο κοινωνικό δίκτυο αναφερόμαστε συνήθως σε ένα σύνολο από οντότητες και πως αυτές αλληλεπιδρούν μεταξύ τους. Συνήθως αναπαριστούμε το κοινωνικό δίκτυο με τη μορφή ενός γραφήματος, στο οποίο οι οντότητες (άνθρωποι ή πράγματα) αποτελούν τους κόμβους και οι αλληλεπιδράσεις (ή σχέσεις) μεταξύ τους αποτελούν τις ακμές. Αρχικά, μελετάμε την ποικιλομορφία των συνδεδεμένων κόμβων σε ένα δίκτυο. Συγκεκριμένα, μελετάμε αν οι κόμβοι τείνουν να συνδέονται με όμοιους κόμβους, το οποίο οδηγεί σε χαμηλή ποικιλομορφία. Επιπλέον μελετάμε την επίδραση των αλγορίθμων πρόβλεψης και σύστασης ακμών σε αυτό το φαινόμενο. Συγκεκριμένα, εξετάζουμε αν οι αλγόριθμοι αυτοί προτείνουν στους κόμβους όμοιους κόμβους για σύνδεση. Προσδιορίζοντας την ομοιότητα, δημιουργώντας κάποια κοινωνικά δίκτυα και εκτελώντας κάποια πειράματα, αποδεικνύουμε πως η ποικιλομορφία στα κοινωνικά δίκτυα είναι πράγματι χαμηλή και πως οι αλγόριθμοι ενισχύουν αυτό το φαινόμενο. Επιπλέον, εξηγούμε γιατί αυτό το φαινόμενο είναι δυσάρεστο. Ειδικότερα, η εργασία αυτή αποτελείται από πέντε κεφάλαια. Στο πρώτο κάνουμε μία εισαγωγή και προσδιορίζουμε το πρόβλημα που μελετάμε. Στο δεύτερο παρουσιάζουμε κάποιους αλγορίθμους πρόβλεψης και σύστασης ακμών που χρησιμοποιούνται σε πειράματα. Στο τρίτο, δημιουργούμε ένα συνθετικό δίκτυο, εκτελούμε κάποια πειράματα και εξάγουμε κάποια συμπεράσματα. Αντίστοιχα, στο τέταρτο δημιουργούμε ένα δίκτυο από πραγματικά δεδομένα, εκτελούμε κάποια πειράματα και εξάγουμε κάποια συμπεράσματα. Τέλος, στο πέμπτο συνοψίζουμε όλη τη διαδικασία που ακολουθήσαμε και τα συμπεράσματα που έχουν προκύψει σχετικά με το πρόβλημα και παρουσιάζουμε κάποιες ιδέες για μελλοντική δουλεία.

Λέξεις Κλειδιά: Ανάλυση κοινωνικών δικτύων, πρόβλεψη συνδέσμων, σύσταση συνδέσμων, ποικιλομορφία

Table of Contents

Chapter 1

Introduction

Social network analysis (SNA) is the process of investigating social structures through the use of networks and graph theory. It characterizes networked structures in terms of nodes (individual people, or things within the network) and the edges, or links (relationships or interactions) that connect them. Examples of social structures commonly visualized through social network analysis include social media networks, friendship and acquaintance networks, business networks, social networks and collaboration graphs. These networks are often visualized through graphs in which nodes are represented as points and vertices are represented as edges.

We analyze different networks in order to study the diversity in them. Low diversity appears between the linked nodes in a network, when most nodes are connected with similar ones. In real life there is also low diversity in human relationships. People tend to be friends with people with their own interests, opinion or beliefs. Low diversity leads people to live in a "world" dominated by their own preferences and the preferences of their friends. This makes them lose the feeling of what is generally in the "world". For this reason they also think that only their view is true and phenomena such as football fanaticism are intensifying. Moreover, low diversity does not allow people to be influenced by different types of music, culture or cuisine, because their “world” has only what they prefer. Finally, the low diversity of linked nodes prevents people from evolving, to fear, or to not know what else is in the world, and not to develop their thinking.

Apart from the problem of low diversity, we are studying some link prediction and recommendation algorithms. Link prediction and link recommendation algorithms are algorithms that have as input the snapshot of a network at a given time and produce as result links between network’s nodes, that don’t exist and are likely to be created in the future. We analyze these algorithms in terms of their predictability and the overall results they suggest. We want to see if they tend to suggest similar nodes in network’s nodes for connection. If this happens, then they intensify the phenomenon of low diversity in the network and make diversity gradually less.

In conclusion, the problem we are studying is the diversity in social networks and if link prediction and link recommendation algorithms affect this. For this reason we create some networks based on synthetic and actual data and count the similarity in them. We use also some link prediction and link recommendation algorithms to see what kind of results they produce. Most of the algorithms are based on the structure of the network and one of them is based on both structure and attributes of the nodes. The whole process is described in the following chapters.

Chapter 2

Link prediction algorithms

In this chapter we describe some link prediction algorithms, that we use in our experiments. In particular, we explain the link prediction problem, analyze the link prediction algorithms, and consider how we can process data for algorithm usage and testing [2]. We also present a special algorithm for link prediction and link recommendation based on supervised random walks in graph [3].

2.1 The link prediction problem

Social networks are dynamic objects. This means that they grow and change quickly over time through the addition of new edges, suggesting the emergence of new interactions. The link prediction problem examines what new edges will appear in a given snapshot of a network in the future. There are many factors that can create a new edge. They are factors that affect the structure of the network, the characteristics of the nodes, the interactions between the nodes and so on.
Some interesting uses of link prediction problem are in the business sector, where they can suggest promising interactions or collaborations that have not yet been utilized between their members. In a different vein, in security, link prediction can help to the problem of monitoring terrorist networks; link prediction in this context allows one to conjecture that particular individuals are working together, even though their interaction has not been directly observed.
We study some link prediction algorithms below, that use the structure of the network and also a link prediction algorithm that also takes into account the attributes of the nodes and the attributes of the edges.

2.2 Methods based on the structure of the network

There are many methods for link prediction. To predict the algorithm the edges to be created in the future for a seed node, it gives a score to each other node. The score is defined differently depending on the method. Nodes with high score are candidates for future edge. In

this subsection we will focus on methods based on the structure of the network, and more specifically, based on node neighborhoods and network paths. All the methods assign a connection weight score(x,y) to pairs of nodes <x,y>, based on the input Graph, and then produce a ranking list with descending order of score(x,y). Thus, they can be viewed as calculating a measure of proximity between nodes x and y, in relation to the topology of the network. Generally, the methods are adapted from techniques used in graph theory and social network analysis.

2.2.1 Methods based on node neighborhoods

For a node x, let D(x) be the set of neighbors of x in the Graph. A number of link prediction methods are based on the idea that two nodes x and y are more likely to form a link in the future, if they have many common neighbors; D(x) and D(y) have large overlap. Common neighbors and Jaccard's coefficient are such kinds of methods.

- **Common neighbors.** In this method score(x,y) is the number of neighbors that x and y have in common:

$$score(x,y) = |D(x) \cap D(y)|$$

- **Jaccard's coefficient.** This is a commonly used similarity metric in information retrieval. It measures the probability that both x and y have feature f, for a feature f, that either x or y has. If "features" are neighbors in Graph, this leads to the measure:

$$score(x,y) = \frac{|D(x) \cap D(y)|}{|D(x) \cup D(y)|}$$

Table 2.2.1 summarizes all the scores of the methods.

2.2.2 Methods based on network paths

Let x, y be nodes from graph G. A number of link prediction methods use shortest-path between x and y to set score. They also refine the notion of shortest-path distance by implicitly considering the ensemble of all paths between two nodes. Katz and SimRank are such kinds of methods.

- **Katz.** Katz defines a measure that directly sums over this collection of shortest-paths, exponentially damped by length to count short paths more heavily. This leads to the measure:

$$score(x,y) = \sum_{l=1}^{\infty} \beta^{l} \cdot |paths < l > x,y|$$

where paths<l>x,y is the set of all length-l paths from x to y. A very small β yields predictions much like common neighbors, since paths of length three or more contribute very little to the summation. There are two variants of
Katz measure: (1) unweighted, in which paths<l>x,y = 1 if x and y have collaborated and 0 otherwise, and
(2) weighted, in which paths<l>x,y is the number of times that x and y have collaborated.

- **SimRank.** SimRank is defined as follows: two nodes are similar to the extent that they are joined to similar neighbors. Numerically, this is specified by defining similarity(x, x) = 1 and

$$similarity(x,y) = \gamma \cdot \frac{\left(\sum_{a \epsilon D(x)} \sum_{b \epsilon D(y)} similarity(a,b)\right)}{|D(x)| \cdot |D(y)|}$$

for some γ ∈ [0, 1]. We then define:

$$score(x,y) = similarity(x,y)$$

Methods	Scores
Common neighbors	$score(x,y) = \|D(x) \cap D(y)\|$
Jaccard's coefficient	$score(x,y) = \|D(x) \cap D(y)\| \,/\, \|D(x) \cup D(y)\|$
Katz	$score(x,y) = \sum_{l=1,\dots,\infty} \beta^l \cdot \|paths\langle l \rangle x,y\|$
SimRank	$score(x,y) = similarity(x,y)$
Graph distance	length of shortest path between x and y

Table 2.2.1*: Summary of methods for link prediction. x,y are network nodes.*

2.3 Algorithm based on structure and attributes

In this subsection we describe an algorithm that uses Supervised Random Walks with Restart for link prediction and link recommendation [3]. The difference with the above methods is that in this case, the structure of the network and characteristics (attributes, features) of the nodes and edges of the network are combined into a unified link prediction algorithm. We can see as attributes of nodes, the user profile information, like age, home town, preferences and as attributes of edges, the interaction information between the nodes, like messages that have been sent between friends in a social network.
To predict new edges for a node, first a function is defined that gives a "strength" value to each edge, based on the attributes of linked nodes and the edge. This function is parameterized by some weights, that the algorithm learns by solving an optimization problem, described in the subsection 2.3.1, and a value is given on each edge using the function with the properly weights. Then a random walk with restart is run from the node of care. The stationary distribution p of the random walk assigns each node a probability p_{node}. Nodes are ordered by p_{node} and top ranked nodes are then predicted as destination nodes of future edges of start node.

2.3.1 The optimization problem

The point is to predict nodes for link for the sourse nodes s, such as d ∈ D, where D are destination nodes and not l ∈ L, where L are nodes that aren't destination nodes. Destination nodes are nodes, that create edge with s in the future. The function that computes a strength a_{uv} for each edge is $f_w(\psi_{uv})$. ψ_{uv} is the feature vector. We aim to set the parameters w of function $f_w(\psi_{uv})$ so that it will assign edge weights a_{uv} in such a way that the random walk will be more likely to visit nodes in D than L, i.e., $p_l < p_d$, for each d ∈ D and l ∈ L. Thus, we define the optimization problem to find the optimal set of parameters w of edge strength function $f_w(\psi_{uv})$ as follows:

$$\min_w F(w) = ||w||^2$$
$$\text{such that}$$
$$\forall\, d \in D, l \in L:\ p_l < p_d$$

where p is the vector of PageRank scores. Note that PageRank scores p_i depend on edge strengths a_{uv} and thus actually depend on $f_w(\psi_{uv})$ that is parameterized by w. The idea here is that we want to find the parameter vector w such that the PageRank scores of nodes in D will be greater than the scores of nodes in L. But this is a "hard" version of the optimization problem as it allows no constraints to be violated. In practice it is unlikely that a solution satisfying all the constraints exist. So we solve instead the optimization problem:

$$\min_{w} F(w) = ||w||^2 + \lambda \sum_{d \in D, l \in L} h(p_l - p_d)$$

where λ is the regularization parameter that trades-off between the complexity (i.e., norm of w) for the fit of the model. Moreover, h(·) is a loss function, that assigns a non-negative penalty according to the difference of the scores $p_l - p_d$. If $p_l - p_d < 0$ then h(·) = 0 as $p_l < p_d$ and the constraint is not violated, while for $p_l - p_d > 0$, also h(·) > 0.

2.4 Evaluation

We now look at how we can process some data to run and test the above algorithms. We use this way of processing in future experiments. Let's assume that we have a snapshot of a social network G = <V, E> at a particular time. V is the set of nodes of G and E is the set of edges between the nodes. If e = <u, v> belongs to E, then e is an edge between u and v, where u and v belong to V.
We randomly separate all the edges of each node into two subsets. The first subset is called training set and we use it to train a link prediction algorithm and the second subset is called test set and we use it to test the results of the prediction. The prediction concerns specific nodes that exist in both training set and test set. Thus, if we give as input to the algorithm the training set, we take as output a list of edges that do not appear in the training set and are expected to be created. With the test set, we can check if the predictions are valid.

Chapter 3

Synthetic network analysis

This chapter refers to the creation and analysis of a synthetic network. Below, we analyze the creation of the network, the similarity that is observed on it and the results of link prediction and link recommendation algorithms.

3.1 Creating synthetic networks

To investigate the problem we first create some networks of synthetic data. Below we analyze what preferential attachment and homophily is and how we combine both of them to create synthetic networks.

3.1.1 Preferential attachment and homophily

Preferential attachment and homophily are among the most recognized mechanisms of network evolution.

Preferential attachment is a model for the development of an existing network. Suppose we have a network with some nodes connected to each other. When a new node "comes" then the probability of attaching to a given node is proportional to it's degree. So nodes that have very high degree, are going to have higher chances.

Homophily is the tendency of individuals to associate and bond with similar others. The presence of homophily has been discovered in a vast array of network studies. More than 100 studies have observed homophily in some form or another and they establish that similarity breeds connection. These include age, gender, class, and organizational role. Individuals in homophilic relationships share common characteristics (beliefs, values, education, etc.) that make communication and relationship formation easier.

3.1.2 Combining Preferential attachment and Homophily

For the creation of the synthetic networks, the simplest would be to create totally random networks in which each new node could connect to any network node in a random manner. But that would not have anything to do with reality. In actual networks nodes tend to be connected to nodes to which many nodes have been connected. It means that "rich" become "richer" and "poor" become "poorer." This can also be ascertained by real life. For example, a scientist is more likely to read a paper that is quite popular than one that is not so well known, or someone to subscribe to a channel on YouTube that is famous and has many subscribers to a channel that is not so well known and has few subscribers. It is therefore important to use the method preferential attachment for creating the synthetic networks. However, we don't just use the preferential attachment method, as it is generally done in the creation of a synthetic network, but we make some modifications, so that we take into account the similarity of the nodes and the homophily. For this reason we use a parameter h for homophily in the creation of the synthetic networks. We do this because we want to control the similarity in the networks. Bellow we explain how parameter h works.

3.1.3 The creation of the networks

The procedure of each network creation is divided into two phases. Initially, we create a network that consists of a small number of linked nodes. Each node has an attribute, let's assume 0 or 1. Then we extend the network with new nodes. When a new node "comes", then the probability of connecting to a given node is not just proportional to the degree of the node, but it is the product of the degree of the node and the probability Pr(newNode, node) **(1)**. This probability is related to the similarity parameter h, described in previous subsection. More specifically, it is equal to h when the new node has the same attribute as the node we are studying, i.e. when they both have attribute 0 or both 1, while equal to 1 if they have a different attribute, one 0 and the other 1 **(2)**.

$$Pr\_connection(node) = d(node) * Pr(newNode, node) \quad \textbf{(1)}$$

where Pr(newNode, node) is described in the equation (2).

$$Pr(newNode, node) = \begin{cases} h, & \text{if attribute(node)= attribute(newNode)} \\ 1\text{-}h, & \text{otherwise} \end{cases} \quad \textbf{(2)}$$

where h takes values from 0 to 1.
If it is equal to 0, then each new node tends to connect to nodes that are not exactly the same. When it is equal to 1, then each new node tends to connect to similar nodes. For 0.5, the similarity does not play any role, so we have simple preferential attachment without homophily. ***Figure 3.1.3.1*** shows an example of a synthetic network with 5 initial nodes, 1000 total nodes, 4 edges per new node and h = 0.7.

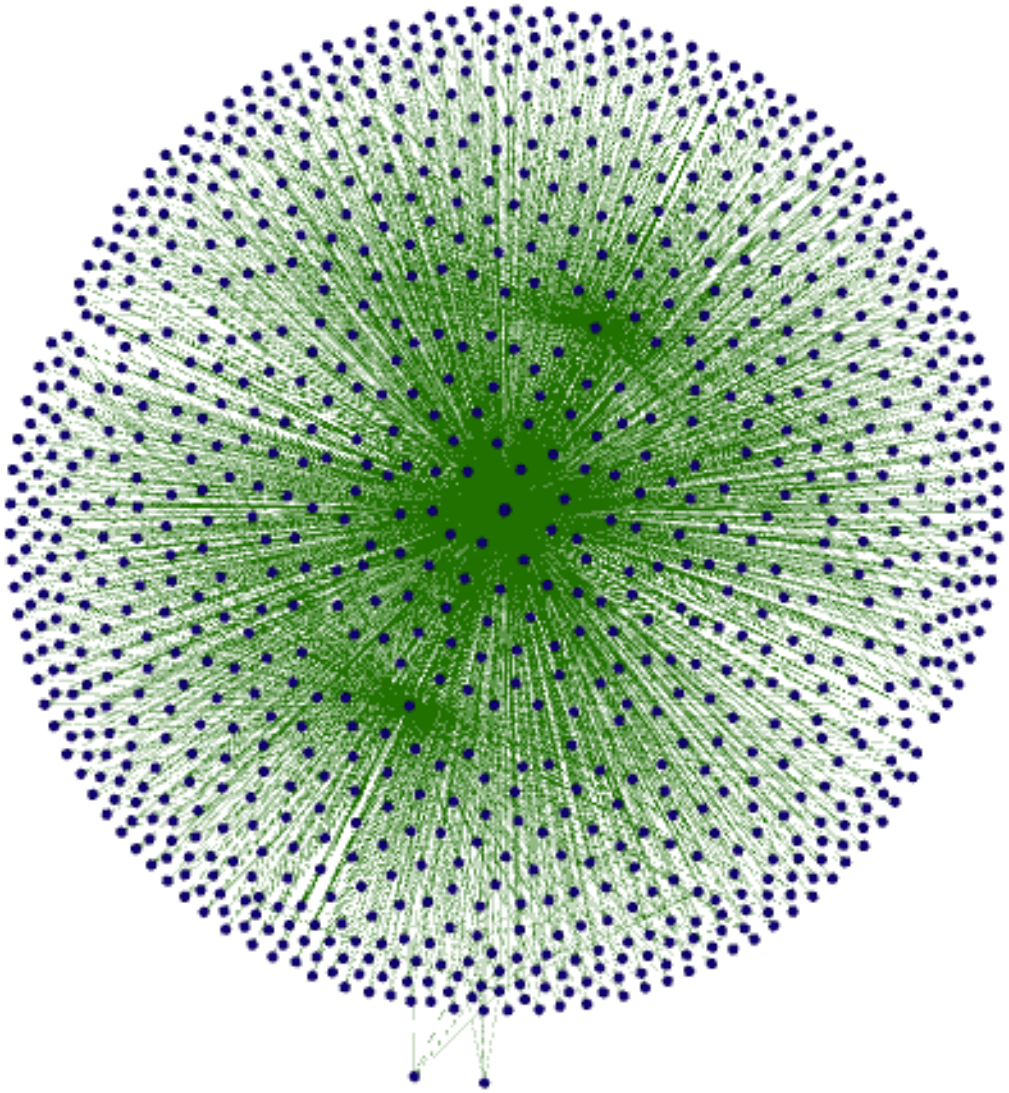


Figure 3.1.3.1*: Synthetic network example with 1000 nodes.*

3.2 Definition of similarity

From the creation of the synthetic network it follows, that each node is characterized by an attribute, 0 or 1. We use these attributes to measure the similarity between the nodes of the synthetic network. But let's first determine how we calculate the similarity.
Let N be the set of nodes belonging to the same neighborhood. The neighborhood of a node is the set with all of his friends. So if u is the node of interest, then N(u) is the set of nodes in the neighborhood of node u. We will now define function Check_Similar **(1)**, that it is equal to 1 if two nodes have same attributes, and zero otherwise.

$$\text{Check\_Similar}(u,v) = \begin{cases} 1, & \text{if attribute}(u) = \text{attribute}(v) \\ 0, & \text{otherwise} \end{cases} \quad \textbf{(1)}$$

where u, v are nodes in the network.

We define as similarity of node u, the sum of the neighbors of u, having the same attribute as u, divided by the number of the neighbors of u **(2)**.

$$\text{Similarity}(u) = \sum_{v \in N(u)} \text{Check\_Similar}(u,v) / |N(u)| \quad \textbf{(2)}$$

3.3 Experiments

In this subsection we calculate the similarity between nodes in the same neighborhood in synthetic networks, and we use also the algorithms from section 2 to check whether link prediction and recommendation algorithms suggest similar nodes for connection.
To be more fair we are not just creating one synthetic network. We create five networks, we calculate the similarity to them, we apply link prediction and recommendation algorithms to them and we get the average of the results.
Bellow we perform four experiments. In the first experiment, we use networks with high value for parameter h, because we want nodes to link with similar ones. We also give same probability to attributes 0 and 1 for each node. In the second experiment, we use networks with low value for parameter h, because we want nodes to link with nodes that aren't similar. Attributes are also given here on nodes with the same probability. In the third experiment, we

use networks with high value for parameter h, because we want nodes to link with similar ones. The difference is that now we give greater probability to attribute 1 than to attribute 0, because we want to see if the size of a group plays role in similarity and link prediction. We define as group the nodes with the same attribute. Finally, in the fourth experiment we use networks with low value for parameter h, because we want nodes to link with different others. We also give here greater probability to attribute 1, than to attribute 0.

The parameters are summarized in ***Table 3.3.1***.

Parameter	Description	Same size – homophily	Same size – no homophily	Different size – homophily	Different size – no homophily
N	Number of nodes	1000	1000	1000	1000
h	homophily	0.8	0.2	0.8	0.2
r	Number of edges per new node	3	3	3	3
p	Relative goup size, for the larger group	0.5	0.5	0.97	0.97

Table 3.3.1*: Parameters used in experiments.*

3.3.1 The same size in attribute groups and high homophily in the network

In the first experiment, we create five synthetic networks with 3 initial nodes, 1000 total nodes and 3 edges per new node per network. Attributes are randomly assigned to nodes with a probability of 50%. Parameter h has value 0.8, because we want nodes to connect to similar ones. Calculating the similarity in the networks and taking the mean, we notice, as we expected, that similarity is very high; it is close to 1 or 1. We observe also that the most of the nodes have very low degree (=3) and only few of them have high degree. ***Figure 3.3.1.1*** shows the average similarity per degree for the five synthetic graphs.
The five synthetic networks are low in diversity, because the linked nodes are similar. The next step is to run the algorithms described in section 2 to check if they find destination nodes and how similar they are. We use Common Neighbors and Jaccard's coefficient from the methods based on neighborhoods and Katz and SimRank from methods based on network paths. We also use the Supervised Random Walks algorithm that combines network structure and node attributes.
First we divide each network into two sets. The first is called training and the second test. The technique to do that is described in section 2.4, of chapter 2. We apply the algorithms using the training set and we get some results. Then, we check if there are common nodes in the

results and also in test sets. If there are such nodes, it means that the link prediction algorithm found nodes that are destination nodes; edges are created in the future. We measure the

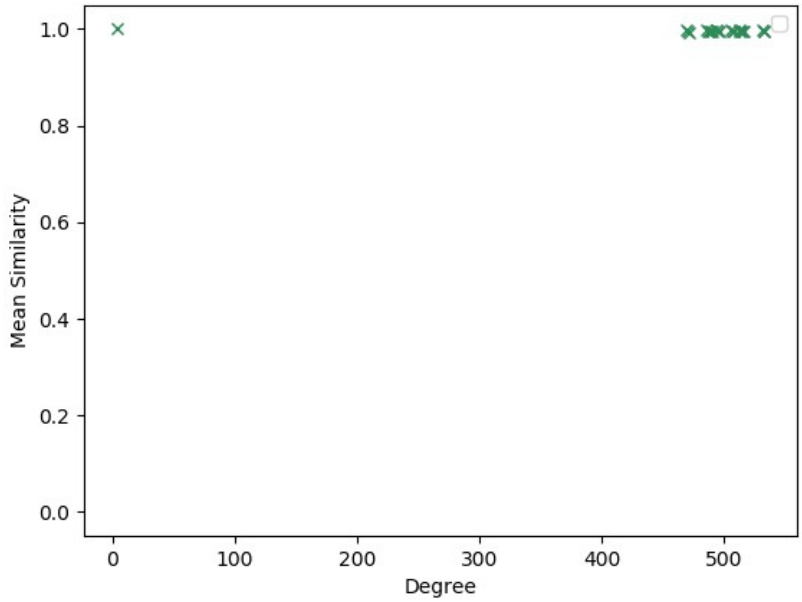


Figure 3.3.1.1*: Mean similarity of five synthetic graphs per degree.*

Jaccard similarity between nodes of care and suggested nodes. We get the mean of the results of each network. For the Supervised Random Walks algorithm, which combines structure and attributes, we need many training examples. So, we take the first three nodes with greater degree, run the algorithm with the training set, get the results and test them with test set.

Algorithms	**Correct Future Edges (success rate)**					**Jaccard Similarity**			
	1st network	2nd network	3rd network	4th network	5th network	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	247 (10%)	285 (11%)	205 (8%)	370 (14%)	221 (8%)	0	0	0.276	99.724
Jaccard's coefficient	247 (10%)	285 (11%)	205 (8%)	370 (14%)	221 (8%)	0	0	0.276	99.724
SimRank	602 (22%)	599 (22.5%)	602 (22.2%)	597 (22%)	631 (22.7%)	0	0	0.218	99.782
Katz	575 (21%)	554 (21%)	578 (21.3%)	588 (21.7%)	601 (21.6%)	0	0	0.244	99.756
Supervised R.W. (avg of 3 nodes)	66.5%	66.7%	99.4%	66.7%	66.7%	24	0	0	76

Table 3.3.1.2*: Link prediction and similarity in results.*

If suggested nodes are destination nodes (future linked nodes) we calculate the similarity between nodes of care and suggested nodes. We get the average of three nodes per network. From the ***Table 3.3.1.2*** we observe that Common Neighbors and Jaccard's coefficient aren't so good at link prediction. SimRank and Katz are much better, while Supervised Random Walks is the best. Similarity in results is high, as we expected, because the graph has high homophily.

But in order to see if the algorithms generally tend to suggest similar nodes for connection, we check beyond the nodes that are actually linked in the future, the overall algorithms results.

Algorithms	**Jaccard Similarity**			
	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	0.226	0.272	78.712	20.79
Jaccard's coefficient	0.226	0.272	78.712	20.79
SimRank	0.14	52.48	47.08	0.3
Katz	0.64	41.68	47.06	0.16
Supervised R.W. (avg of 3 nodes)	24	0	0	76

Table 3.3.1.3*: Link prediction and similarity to overall results.*

Table 3.3.1.3 shows the average of the results of the five synthetic networks for CommonNeighbors, Jaccard's coefficient, SimRank, Katz and Supervised Random Walks algorithms. We notice that all suggested results are quite similar to nodes of care. Moreover, Common Neighbors and Jaccard's coefficient suggest more similar results than SimRank and Katz. For Supervised Random Walks algorithm we take into account the suggested nodes with the highest PageRank score. We observe that the average similarity of three nodes is high.

Now we can conclude that if we have a network with very similar linked nodes and neighborhoods with low diversity, then link prediction and recommendation algorithms also suggest similar nodes for connection. This means that this phenomenon of low diversity is intensifying and the nodes continue to be associated with similar ones, reducing diversity more and more.

3.3.2 The same size in attribute groups and low homophily in the network

In the second experiment, we create five synthetic networks with 3 initial nodes, 1000 total nodes and 3 edges per new node, per network. Attributes are randomly assigned to nodes with a probability of 50%. Parameter h has value 0.2, because we want the nodes to connect with nodes that are not similar. So we create networks with high diversity between linked nodes. Calculating the similarity in the networks and taking the mean, we notice, as we expected, that similarity is very low; it is close to 0 or 0. We observe also that the most of the nodes have very low degree (=3) and only few of them have high degree. ***Figure 3.3.2.1*** shows the average similarity per degree for the five synthetic graphs.

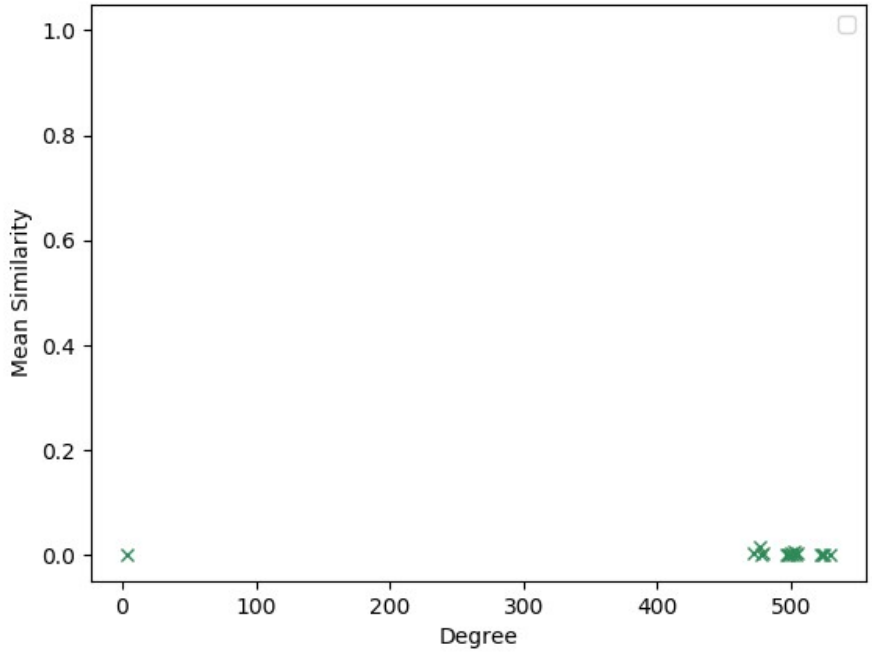


Figure 3.3.2.1*: Mean similarity of five synthetic graphs per degree.*

The next step is to execute the algorithms described in section 2. We use Common Neighbors and Jaccard's coefficient from the methods based on neighborhoods and Katz and SimRank from methods based on network paths. We also use the Supervised Random Walks algorithm that combines network structure and node attributes.
First we divide each network into training and test set. We run the algorithms using the training set and we get some results. Then we check whether there are common nodes in results and also in test sets. If there are such nodes, it means the link prediction found nodes that are destination nodes; edges are created in future. We calculate the Jaccard similarity between nodes of care and suggested nodes. We get the mean of the results. For the Supervised Random Walks algorithm, that combines structure and attributes, we need many training examples. So we take the first three nodes with greater degree, run the algorithm with the training set, get the results and test them with test set. If suggested nodes are destination nodes (future linked nodes) we count the similarity between them and the nodes we study.

Algorithms	Correct Future Edges (success rate)					Jaccard Similarity			
	1st network	2nd network	3rd network	4th network	5th network	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	132 (5%)	165 (6%)	137 (5.1%)	219 (8%)	158 (5.7%)	99,44	0	0	0,56
Jaccard's coefficient	132 (5%)	165 (6%)	137 (5.1%)	219 (8%)	158 (5.7%)	99,44	0	0	0,56
SimRank	630 (22.8%)	615 (22.2%)	607 (22.4%)	656 (23.6%)	635 (23.2%)	99.86	0.05	0	0.09
Katz	603 (21,8%)	594 (21.4%)	589 (21.7%)	647 (23.3%)	620 (22.7%)	99.86	0.05	0	0.09
Supervised R.W. (avg of 3 nodes)	99.5%	99.7%	99.7%	66.5%	33.3%	93	0	0	7

***Table 3.3.2.2**: Link prediction and similarity in results.*

From the ***Table 3.3.2.2*** we observe that Common Neighbors and Jaccard's coefficient aren't so good at link prediction. SimRank and Katz are better, while Supervised Random Walks is the best. Similarity in results is low, as we expected, because the graph has low homophily.

But in order to see if the algorithms generally tend to suggest similar nodes for connection, we check beyond the nodes that are actually linked in the future, the overall algorithms results.

Algorithms	Jaccard Similarity			
	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	0.06	0	21.12	78.82
Jaccard's coefficient	0.06	0	21.12	78.82
SimRank	0.05	44.28	55.8	0.14
Katz	0.06	44.3	55.5	0.14
Supervised R.W. (avg of 3 nodes)	93	0	0	7

***Table 3.3.2.3**: Link prediction and similarity to overall results.*

Table 3.3.2.3 shows the average of the results of the five synthetic networks for CommonNeighbors, Jaccard's coefficient, SimRank, Katz and Supervised Random Walks algorithms. We notice that all suggested results are quite similar to nodes of care, although the network has low homophily. Moreover, for Supervised Random Walks algorithm we take the average similarity of the mean of the first three nodes in the five synthetic graphs, considering the proposed higher ranked PageRank nodes. We notice that the suggested nodes aren't similar to nodes we are interested in. That is because the algorithm takes into account the attributes of the nodes, and not just the structure of the network. The algorithm also is trained with the attributes and learns to predict nodes like the training examples. In our case the connected nodes aren't similar, so the algorithm suggests such kind of nodes for connection.

We conclude that if we have a network with nodes that have approximately the same number of attributes 0 and 1, and the connected nodes are not similar, then the algorithms that take into account only the structure of the network suggest similar nodes for connection, while the algorithm taking into consideration the structure and the attributes suggests nodes which are not similar.

3.3.3 Different size in attribute groups and high homophily in the network

In the third experiment, we create five synthetic networks with 3 initial nodes, 1000 total nodes and 3 edges per new node, per network. Parameter h has value 0.8, because we want the nodes to connect to nodes that are quite similar. Attribute 1 is given to nodes with a probability of 97%, while attribute 0 is given with a probability of 3%. Therefore, the networks are low in diversity between linked nodes and the group with nodes having attribute 1 is larger than the group with the nodes having attribute 0. ***Figure 3.3.3.1*** shows the mean similarity per degree for our five synthetic graphs. We notice that, as we expected, the similarity is very high, close to 1 or 1.

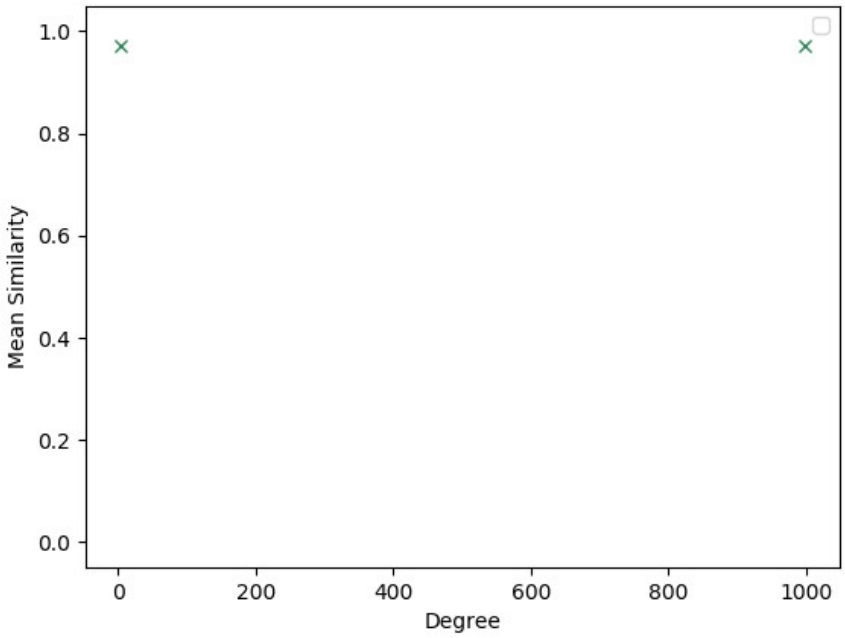


Figure 3.3.3.1*: Mean similarity of five synthetic graphs per degree.*

But let's also calculate the similarity separately for each group. ***Figure 3.3.3.1(a)*** shows the mean similarity for the group with nodes having attributes 1 and ***Figure 3.3.3.1(b)*** shows the mean similarity for the group with nodes having attribute 0. Let's define the group of Figure 3.3.3.1(a) as group 1 and the group of Figure 3.3.3.1(b) as group 0. We observe that group 1, the larger group, has great similarity, while the group 0 has very low similarity almost 0,

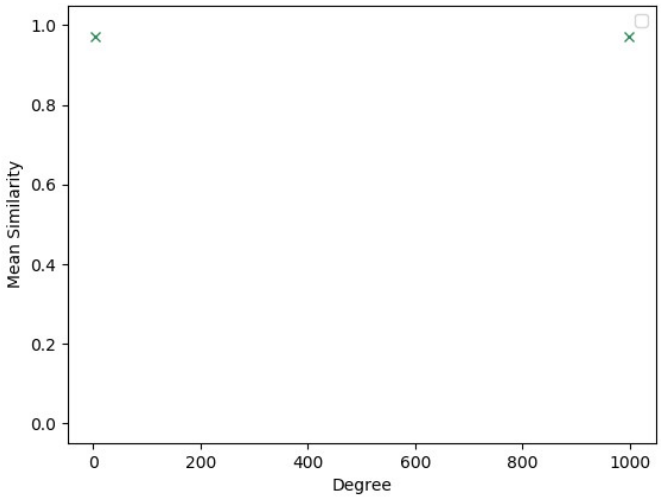


Figure 3.3.3.1(a)*: Mean similarity of five synthetic graphs per degree for nodes with attribute 1.*

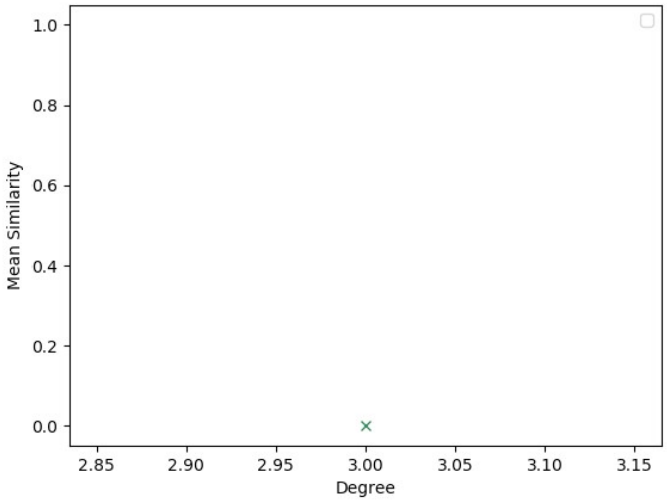


Figure 3.3.3.1(b)*: Mean similarity of five synthetic graphs per degree for nodes with attribute 0.*

although the h parameter has a high value that gives similar nodes greater probability for connection. This is because we take also into account the degree of a node except the h parameter for connection; nodes with attribute 1 are more, they have high degree and force nodes with attribute 0 to connect with them although they are not similar.
The next step is to execute the algorithms described in section 2. Following the same logic with the above experiments, we use Common Neighbors and Jaccard's coefficient from the methods based on neighborhoods and Katz and SimRank from methods based on network paths. We also use the Supervised Random Walks algorithm that combines the structure of the network and the attributes of the nodes.
We again separate each network into training and test sets. We run the algorithms using the training set and we get some results. Then, we check whether there are common nodes in results and also in test sets. If there are such nodes, it means that the link prediction found nodes that are destination nodes; edges are created in future. We calculate Jaccard similarity between nodes of care and suggested nodes. We get the mean of the results. The difference is that we also calculate now the average similarity for each group separately.

Algorithms	Correct Future Edges (success rate)					Jaccard Similarity			
	1st network	2nd network	3rd network	4th network	5th network	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	497 (18.3%)	138 (5.1%)	485 (18.1%)	510 (18.9%)	219 (8.1%)	3.16	0	0.22	96.62
Jaccard's coefficient	497 (18.3%)	138 (5.1%)	485 (18.1%)	510 (18.9%)	219 (8.1%)	3.16	0	0.22	96.62
SimRank	623 (23%)	620 (22.8%)	609 (22.7%)	616 (22.8%)	630 (23.2%)	2.94	0	0.64	96.74
Katz	596 (22%)	590 (21.7%)	582 (21.7%)	592 (22%)	615 (22.7%)	2.96	0	0.32	96.72
Supervised R.W. (avg of 3 nodes)	97.16%	65.7%	65.4%	65.5%	2.4%	38	0	0	62

***Table 3.3.3.2**: Link prediction and similarity in results.*

Algorithms	Jaccard Similarity			
GROUP 0	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	3.16	0	0	0
Jaccard's coefficient	3.16	0	0	0
SimRank	2.94	0	0	0
Katz	2.96	0	0	0

GROUP 1	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	0	0	0.22	96,62
Jaccard's coefficient	0	0	0.22	96,62
SimRank	0	0	0.64	96,74
Katz	0	0	0.32	96.72

***Table 3.3.3.3**: Link prediction and similarity in results for groups 0 & 1.*

Table 3.3.3.2, shows the results of Common Neighbours, Jaccard's coefficient, SimRank, Katz and Supervised Random Walks algorithms, for the global network. For the Supervised Random Walks algorithm, which combines structure and attributes, we need many training examples. So we get again the first three nodes with greater degree, run the algorithm with the training set, get the results and test them with test set. If the suggested nodes are destination nodes (future linked nodes), we calculate the similarity between them and the nodes we study. We observe that in general the link prediction works well, and finds destination nodes that are similar to nodes we study. This is because the most of the network consists of nodes with attribute 1.

Table 3.3.3.3, shows the results separately for each group, 0 & 1. For the Supervised Random Walks algorithm, we don't count the similarity for each group, because we only use the first three nodes of each network for prediction, since they have many training examples. We notice that link prediction works almost perfect for group 1 and suggests nodes that are very similar to nodes of care. For the group 0, link prediction finds nodes, but they don't have attribute 0, because the most nodes with attribute 0 aren't connected with similar nodes.

But let's now see beyond the nodes that are destination nodes, the overall results of the algorithms.

Table 3.3.3.4 and ***Table 3.3.3.5***, show the similarity for groups 0, 1 together and separately, taking into account the overall results of the algorithms. We observe that the similarity for group 1 is high and the similarity for group 0 is low, but better than before.

We conclude that if we have a network with more nodes having attribute 1, than nodes having attribute 0, and the connected nodes are similar, then the group of nodes 1 is very similar, while the group of nodes 0 has low similarity. This means that the size of the group is important for network's similarity. In this network, link prediction works well and finds nodes that are actually destination nodes, very similar to group 1 and not similar to group 0. Finally, the algorithms that take into account only the structure of the network generally suggest similar nodes for connection, very similar nodes to nodes with attribute 1 and also quite similar to them with attribute 0, although the similarity in group 0 is low. The Supervised Random Walks algorithm suggests nodes, which are quite similar to nodes of care.

Algorithms	Correct Future Edges (success rate)					Jaccard Similarity			
	1st network	2nd network	3rd network	4th network	5th network	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	497 (18.3%)	138 (5.1%)	485 (18.1%)	510 (18.9%)	219 (8.1%)	0.1	2.66	91.68	5.56
Jaccard's coefficient	497 (18.3%)	138 (5.1%)	485 (18.1%)	510 (18.9%)	219 (8.1%)	0.1	2.66	91.68	5.56
SimRank	623 (23%)	620 (22.8%)	609 (22.7%)	616 (22.8%)	630 (23.2%)	0.1	2.74	94.96	2.2
Katz	596 (22%)	590 (21.7%)	582 (21.7%)	592 (22%)	615 (22.7%)	0.1	2.72	94.8	2.38
Supervised R.W. (avg of 3 nodes)	97.16%	65.7%	65.4%	65.5%	2.4%	38	0	0	62

***Table 3.3.3.4**: Link prediction and similarity to overall results.*

Algorithms	Jaccard Similarity			
GROUP 0	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	0.1	2.66	0	0
Jaccard's coefficient	0.1	2.66	0	0
SimRank	0.1	2.74	0	0
Katz	0.1	2.72	0	0

GROUP 1	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	0	0	91.68	5.56
Jaccard's coefficient	0	0	91.68	5.56
SimRank	0	0	94.96	2.2
Katz	0	0	94.8	2.38

***Table 3.3.3.5**: Link prediction and similarity to overall results, for groups 0 & 1.*

3.3.4 Different size in attribute groups and low homophily in the network

In the fourth experiment, we create five synthetic networks with 3 initial nodes, 1000 total nodes and 3 edges per new node, per network. The parameter h has value 0.2, because we want the nodes to connect with nodes that are not similar. Attribute 1 is given to nodes with a probability of 97%, while attribute 0 is given with a probability of 3%. Thus, the networks are high in diversity between linked nodes and the group with nodes having attribute1 is larger than the group having attribute 0.

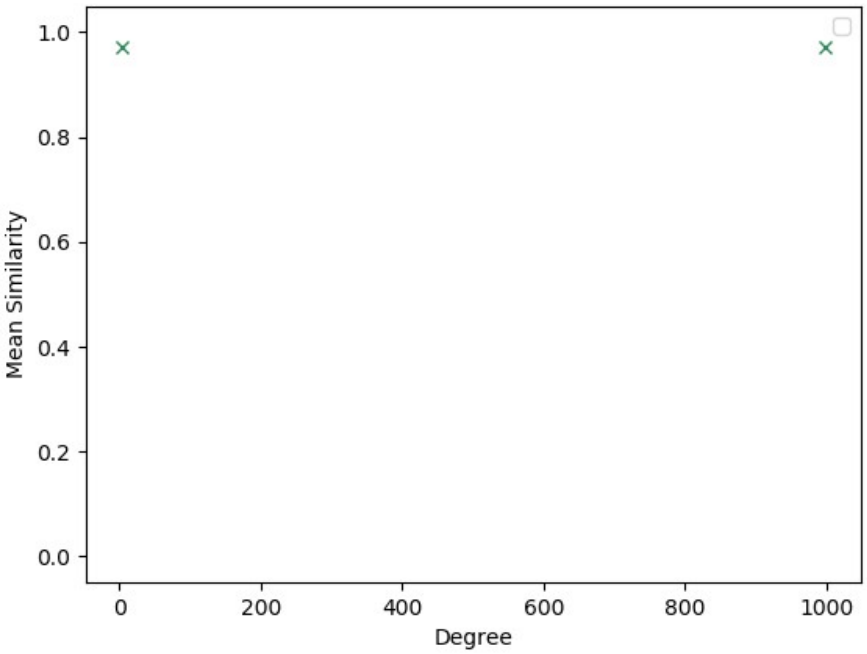


Figure 3.3.4.1*: Mean similarity of five synthetic graphs per degree.*

Figure 3.3.4.1 shows the average similarity per degree for the five synthetic graphs. We observe that the similarity is very high, close to 1 or 1, although the parameter h has a very low value. This is, because also in this case the size of the group affects the similarity. But let's also count the similarity for every group separately. ***Figure 3.3.4.1(a)*** shows the mean similarity for the group with nodes having attributes 1 (group 1) and ***Figure 3.3.4.1(b)*** shows the mean similarity for group with nodes having attribute 0 (group 0). We can see that the group 1 has high similarity notwithstanding it should be low, while group 0 has low similarity. This is because as in the third experiment, parameter h isn't important when the size of a group is much bigger than another group on the network. Once again, we check whether link prediction works well and which results produce link prediction and link recommendation algorithms in general. We again seperate each network into training and test set. We run the algorithms using the training set and we get some results. Then, we check if there are common nodes in the results and also in test sets. If there are such nodes, it means that the link prediction found nodes that are destination nodes; edges are created in future. We calculate Jaccard similarity between nodes of care and suggested nodes. We get the mean of the results. The difference is that now we count the average similarity also for each group separately.

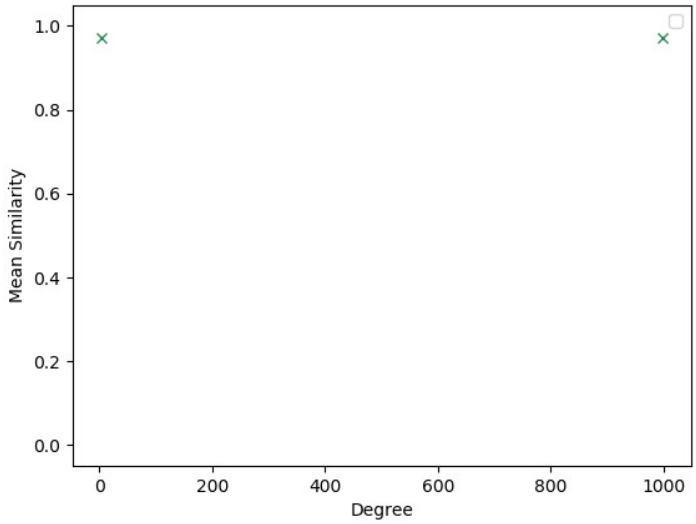


***Figure 3.3.4.1(a)**: Mean similarity of five synthetic graphs per degree for nodes with attribute 1.*

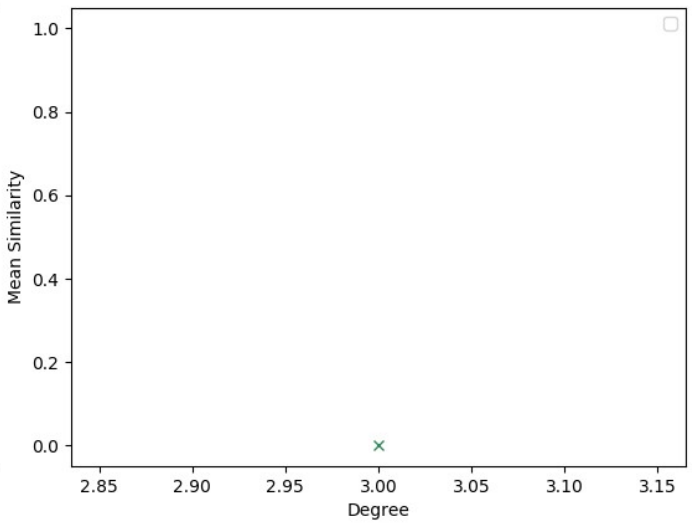


***Figure 3.3.4.1(a)**: Mean similarity of five synthetic graphs per degree for nodes with attribute 0.*

We note that the link prediction works well. Finds nodes that are destination nodes. These nodes are similar to group 1, as we expected, because nodes belonging to group 1 tend to connect with similar ones. On the other hand, the suggested destination nodes for group 0 are not similar and this is, because nodes of group 0 are connected with nodes that are not similar.

Table 3.3.3.2, shows the results of Common Neighbours, Jaccard's coefficient, SimRank, Katz and Supervised Random Walks algorithms, for the global network. For the Supervised Random Walks algorithm, which combines structure and attributes, we need many training examples. So we get again the first three nodes with greater degree, run the algorithm with the training set, get the results and test them with test set. If the suggested nodes are destination nodes (future linked nodes), we calculate the similarity between them and the nodes we study. We observe that in general the link prediction works well, and finds destination nodes that are similar to nodes we study. This is because the most of the network consists of nodes with attribute 1.

Table 3.3.3.3, shows the results separately for each group, 0 & 1. For the Supervised Random Walks algorithm, we don't count the similarity for each group, because we only use the first three nodes of each network for prediction, since they have many training examples. We notice that link prediction works almost perfect for group 1 and suggests nodes that are very similar to nodes of care. For the group 0, link prediction finds nodes, but they don't have attribute 0, because the most nodes with attribute 0 have low homophily.

But let's now see beyond the nodes that are destination nodes, the overall results of the algorithms.

Table 3.3.4.4 and ***Table 3.3.4.5***, show the similarity for groups 0, 1 together and separately, taking into account the overall results of the algorithms. We observe that the similarity for group 1 is high and the similarity for group 0 is low, but better than before.

Algorithms	Correct Future Edges (success rate)					Jaccard Similarity			
	1st network	2nd network	3rd network	4th network	5th network	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	970 (35.9%)	439 (16.1%)	257 (9.4%)	294 (11.1%)	342 (12.4%)	3.2	0	0.06	96.74
Jaccard's coefficient	970 (35.9%)	439 (16.1%)	257 (9.4%)	294 (11.1%)	342 (12.4%)	3.2	0	0.06	96.74
SimRank	615 (22.7%)	632 (23.1%)	644 (23.5%)	586 (22.2%)	617 (22.4%)	2.78	0	0.16	97.06
Katz	588 (21.7%)	605 (22.1%)	608 (22.1%)	562 (21.3%)	584 (21.2%)	2.84	0	0.16	97
Supervised R.W. (avg of 3 nodes)	97.5%	65.5%	65.7%	66%	66%	24	0	0	76

***Table 3.3.4.2**: Link prediction and similarity in results.*

Algorithms	Jaccard Similarity			
GROUP 0	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	3.2	0	0	0
Jaccard's coefficient	3.2	0	0	0
SimRank	2.78	0	0	0
Katz	2.84	0	0	0

GROUP 1	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	0	0	0.06	96.74
Jaccard's coefficient	0	0	0.06	96.74
SimRank	0	0	0.16	97.06
Katz	0	0	0.16	97

***Table 3.3.4.3**: Link prediction and similarity in results for groups 0 & 1.*

Algorithms	Correct Future Edges (success rate)					Jaccard Similarity			
	1st network	2nd network	3rd network	4th network	5th network	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	970 (35.9%)	439 (16.1%)	257 (9.4%)	294 (11.1%)	342 (12.4%)	0.12	2.7	90.2	6.96
Jaccard's coefficient	970 (35.9%)	439 (16.1%)	257 (9.4%)	294 (11.1%)	342 (12.4%)	0.12	2.7	90.2	6.96
SimRank	615 (22.7%)	632 (23.1%)	644 (23.5%)	586 (22.2%)	617 (22.4%)	0.1	2.68	94.22	3
Katz	588 (21.7%)	605 (22.1%)	608 (22.1%)	562 (21.3%)	584 (21.2%)	0.1	2.72	94.98	2
Supervised R.W. (avg of 3 nodes)	97.5%	65.5%	65.7%	66%	66%	24	0	0	76

***Table 3.3.4.4**: Link prediction and similarity to overall results.*

Algorithms	Jaccard Similarity			
GROUP 0	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	0.12	2.7	0	0
Jaccard's coefficient	0.12	2.7	0	0
SimRank	0.1	2.68	0	0
Katz	0.1	2.72	0	0

GROUP 1	0 (%)	0 – 0.5 (%)	0.5 – 1 (%)	1 (%)
Common Neighbors	0	0	90.2	6.96
Jaccard's coefficient	0	0	90.2	6.96
SimRank	0	0	94.22	3
Katz	0	0	94.98	2

***Table 3.3.4.5**: Link prediction and similarity to overall results, for groups 0 & 1.*

We conclude that if we have a network with more nodes with attribute 1, than nodes with attribute 0, and the h parameter has low value and force nodes to connect to others that aren't similar, then the similarity in the network is high. In particular, the size of a group affects similarity and for this reason group 1 of large size has high similarity, while group 0 of small size has low similarity. Link prediction works well on this network and the link prediction and link recommendation algorithms increase the similarity for another time suggesting similar nodes for connection.

In this subsection we performed five experiments. From them we saw that the size of the group of nodes with the same attribute, are important for similarity. The larger group is always similar and affects the small group. We have also noticed that the algorithms that take into account only the structure of the network tend to suggest similar nodes for connection, regardless of the network. This reduces diversity in neighborhoods more and more. On the other hand, the Supervised Random Walks algorithm, which also takes into account the attributes of the nodes, is trained by the network. Therefore, it suggests similar nodes for connection, if the linked nodes on the network are similar and it suggests not similar nodes for connection, if linked nodes on the network are not similar. If the network contains neighborhoods with low diversity, then most link prediction and recommendation algorithms intensify this phenomenon. But this happens even if diversity is high on neighborhoods of the network. So if, for example, we have a network consists of human entities and their relationships, then most algorithms suggest to people other people, who are similar to them. Similarity can be defined in many different ways, such as preferences or views. In these cases, predictions of the algorithms lead to neighborhoods of people with low diversity. This means that people with the same preferences or opinions live in "small worlds" and lose the feeling of what is generally in the "whole world".

In the next chapter we create and analyze an actual data network to see what happens in reality.

Chapter 4

Actual network analysis

This chapter refers to the analysis and experiments performed in an actual network. Below, we analyze the creation of the network, the resulting statistical data, the similarity that is observed and how it is related to the problem we are studying.

4.1 Actual dataset

Actual data was extracted from the data set provided by Yelp. Yelp is a business directory service and crowd-sourced review forum, and a public company with the same name that is headquartered in San Francisco, California. The company develops, hosts and markets the Yelp.com website and the Yelp mobile app. The Yelp data set is a subset of businesses, reviews, and user data for use in personal, educational, and academic purposes.

4.1.1 Pre-processing actual data

The yelp dataset contains JSON files like business.json, review.json, user.json, checkin.json, tip.json and photo.json. We use in pre-processing, **business.json** file, which contains business data including location data, attributes and categories of businesses. We use also, **review.json** file, which contains full review text data including the user_id that wrote the review and the business_id the review is written for and **user.json** file, which contains user data, including the user's friend mapping and all the metadata associated with the user. Based on this information we create a social network as described below.
Initially we select businesses from the business.json file. For local purpose, we are studying businesses that are located in "Urbana" or "Champaign" city and are "Restaurants". Another criterion for the businesses we choose, is to include one or more of the categories "American (Traditional)", "American", "Mexican", "Italian", "Chinese", "Japanese", "Korean", "Indian" and "Greek". Then we select the users who have made reviews of these businesses based on review.json file and their friends. Finally, we check if users' friends have reviewed a business that is restaurant and contains one or more of the above categories. The entire pre-processing is shown in ***Figure 4.1.1***.

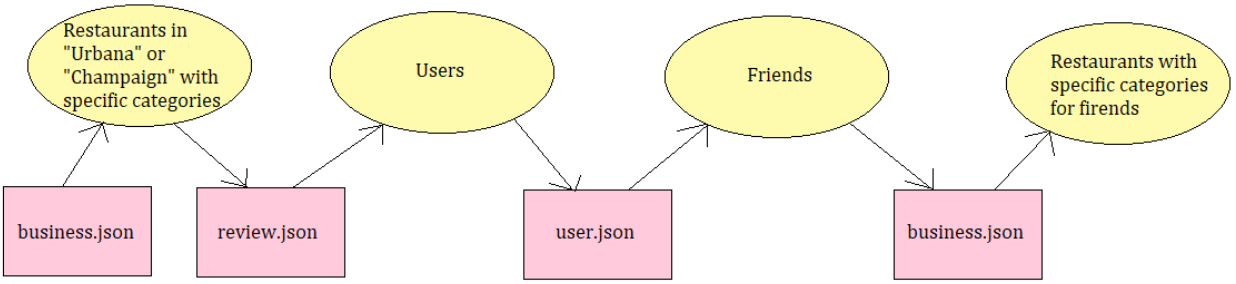


***Figure 4.1.1.1**: Pre-processing of actual data.*

4.1.2 Actual data network

A subset of users emerges from the pre-processing. Among these users there are some friendships. If we consider users as nodes and associations between them as edges, we can create a social network. This network consists of 270,355 nodes and 385,001 edges. The nodes (users) for whom we create associations with other users are 4.314. Also, if we define as indegree the number of edges entering a node and as outdegree the number of edges that exiting a node, then we notice that the nodes have an average of indegree 1 and outdegree 89. We therefore find that we have a fairly sparse network. More features for the network are shown in the table of ***Table4.1.2***.

Seed nodes	4,314
Nodes	270,355
Edges	385,001
Max Outdegree	6,772
Min Outdegree	1
Avg Outdegree per node	89
Max Indegree	148
Min Indegree	0
Avg Indegree per node	1

***Table 2**: Statistical data of the network.*

We can also see the relationship between the number of nodes per outdegree on a logarithmic scale in the graph of ***Figure 4.1.3***. As we excepted, a power-law is observed from the distribution, where the most nodes have small outdegree and just a few nodes have great outdegree.

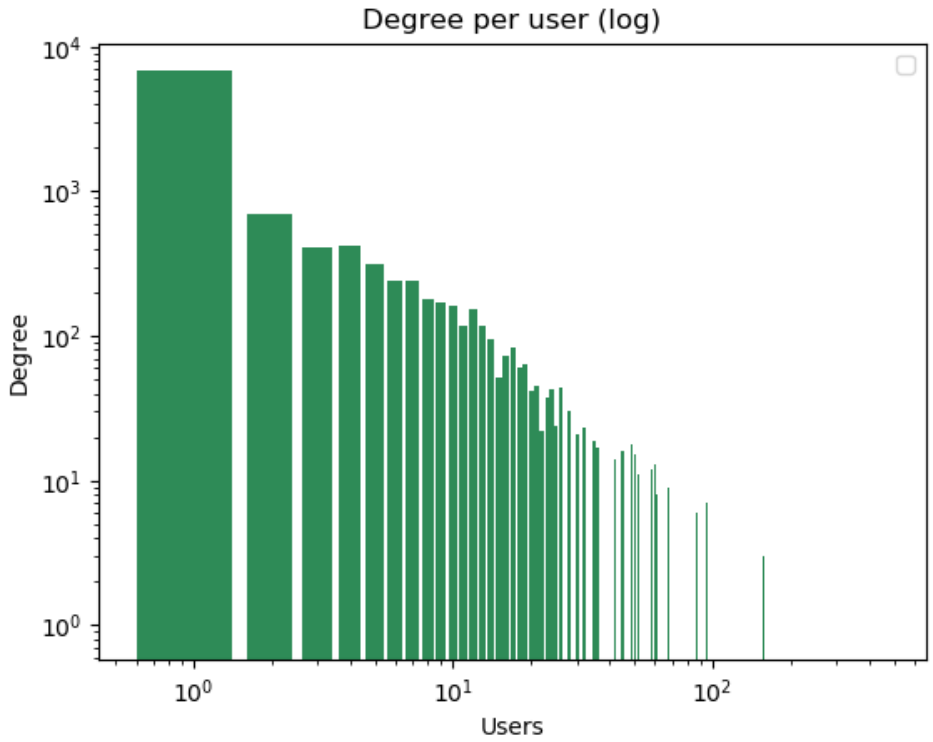
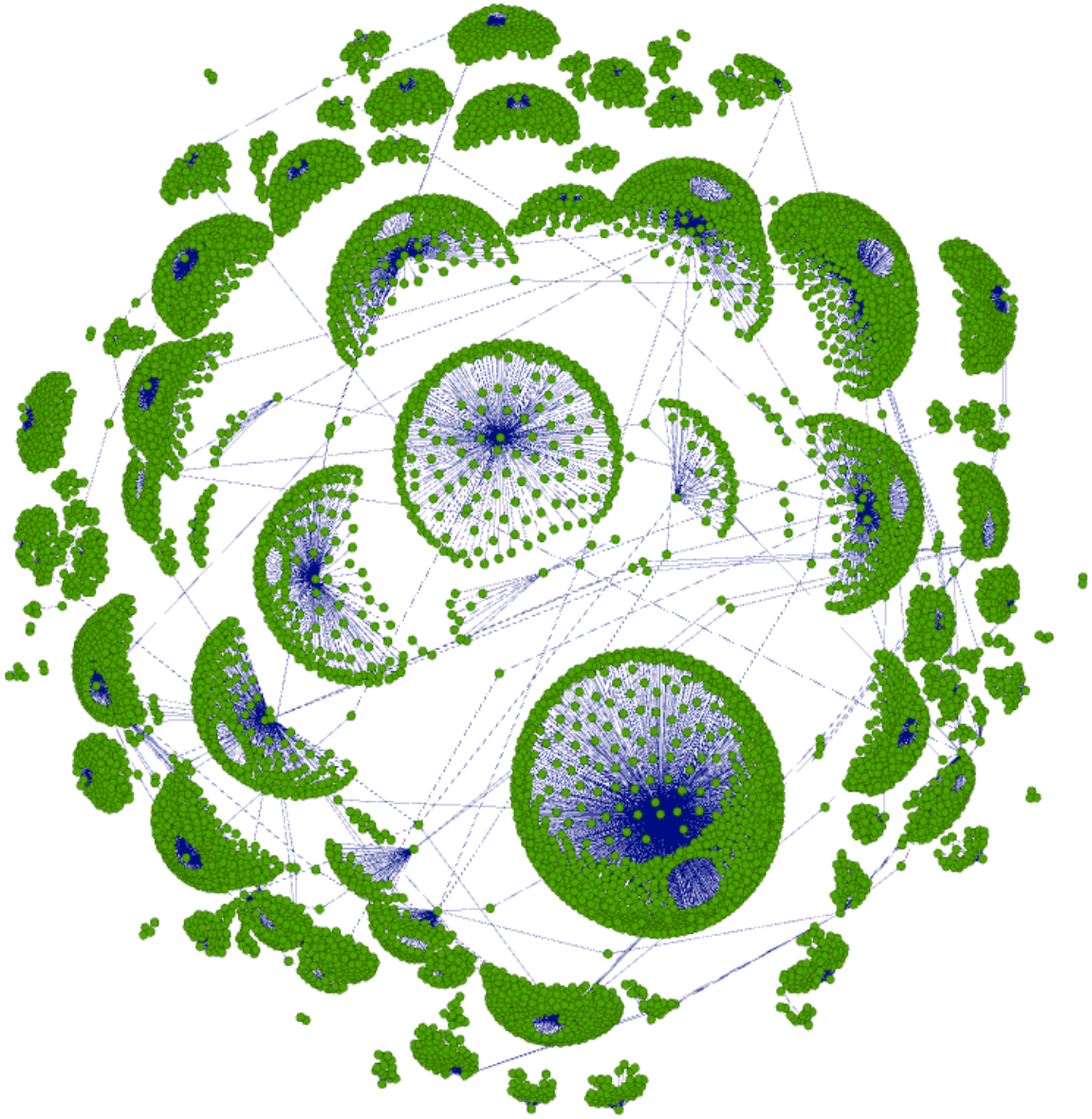


***Figure 4.1.3**: Graph with assiciation between outdegree and number of nodes.*



Subset of the network. *Number of nodes: 8,021. Number of edges: 8,010.*

4.2 Definition of similarity

As can be seen from the pre-processing, described above in section 4.1.1, each node is characterized by some preferences in restaurant types, resulting from the reviews it has made in the respective restaurants. The categories of restaurants we study, and that characterize our network users, are “American (Traditional)”, “American”, “Mexican”, “Italian”, “Chinese”, “Japanese”, “Korean”, “Indian” and “Greek”. Thus, each user has one or more of the above categories derived from their reviews and declare their preferences.

It is a good time to define the similarity, that we use later to make some measurements and produce some results.

Let N be the set of nodes belonging to the same neighborhood. The neighborhood of a node is all of its friends. So if u is the node of interest, then N(u) is the set of nodes in the neighborhood of node u. Let us now define as C the set of restaurant categories reviewed by a node. Then, C(u) is the set of categories of restaurants, that u has reviewed. Based on these and using the Jaccard similarity, we define the similarity between two nodes as JS(u,v) **(1)**

$$JS(u,v) = \frac{|C(u) \cap C(v)|}{|C(u) \cup C(v)|} \quad \textbf{(1)}$$

where,

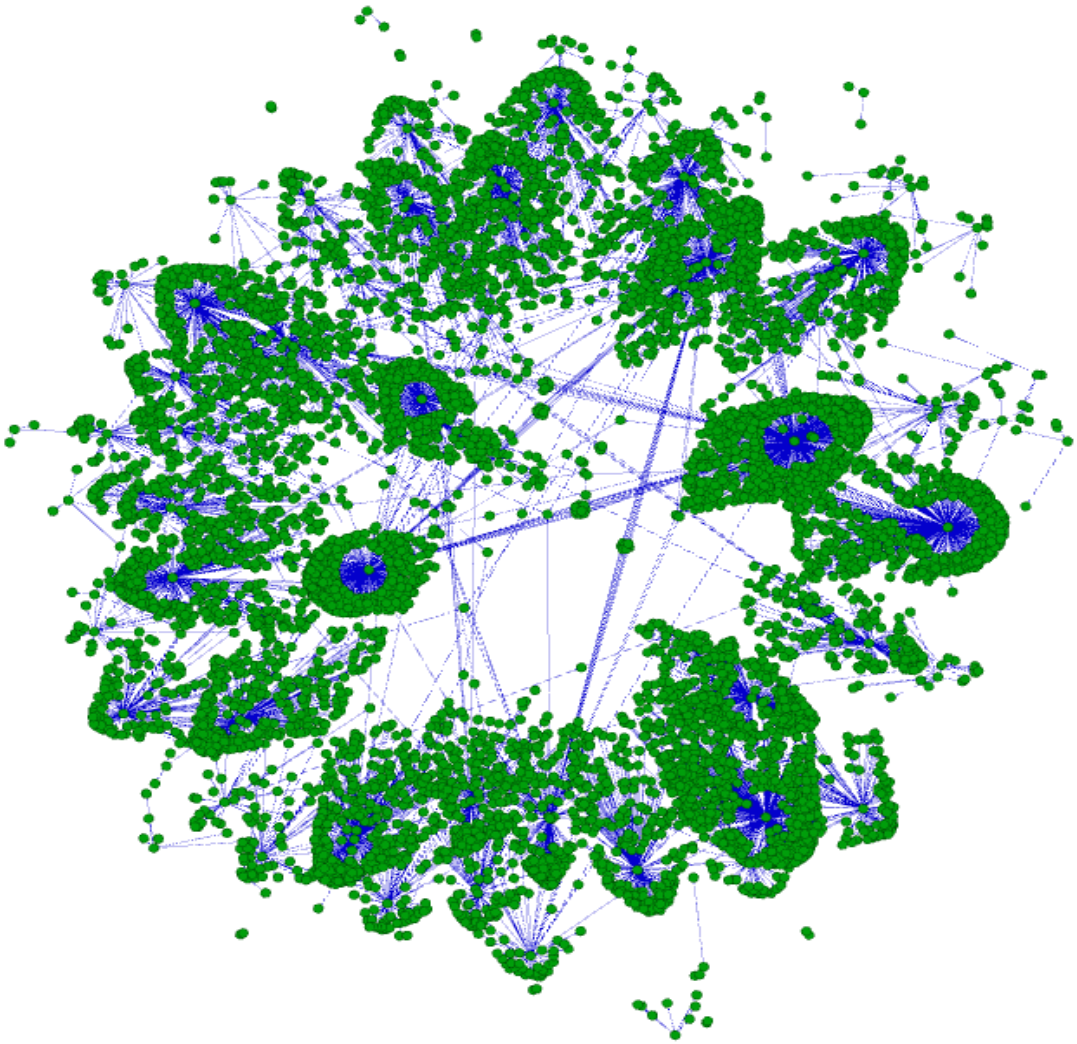
- v is a node and v∈N(u),
- |C(u)∩C(v)| is the number of same categories of restaurants, that u and v have reviewed,
- |C(u)∪C(v)| is the number of distinct categories of restaurants, that u and v have reviewed.

4.3 Creating a random network

Another important thing to do, which we use in later measurements, is to show how a random network is created. With this network, we see later if the edges created between the nodes affect the final results. So we can be sure of the metrics and the conclusions that we make.
In order to create a random network, we follow the procedure below.

- For each node that interests us, we find its outdegree, that is the set of nodes it connects to, its friends.
- Then randomly select as many nodes as its outdegree, to be the new nodes with which it connects, its new friends.

Thus, we have a random network, with the same statistics as the original network, with the same nodes, which have the same characteristics but are connected in a different way.



***Subset of the random network**. Number of nodes: 7,454. Number of edges: 7,525.*

4.4 Experiments

In this subsection, we find the similarity in the actual network and we compare this similarity to the similarity in the random network. We also run some link prediction algorithms and analyze the results.

4.4.1 Actual and random network measurements

In subsection 4.1.2, we created a network of actual data. In this subsection, we use this network for some measurements. The network consists of 385,001 edges and 270,355 nodes. From these nodes we want to find the similarity between the initial nodes(4,314) and their friends(266,041). We also create a random network, as described in subsection 4.3. We use this to compare the results of similarity and to be sure that users tend to connect with similar nodes and it doesn't happen at random. To be fair, we don't calculate similarity only for each node alone, but we also calculate similarity for each degree in the network. In particular, we first calculate similarity for each node of interest and then we calculate similarity for each degree in the network, by taking the average of nodes with same degree.

It is important to mention that we calculate the similarity between a node and one of its neighbors as we described in subsection 4.2, but we define the similarity of a node with its whole neighborhood as the average of the similarities of node with each neighbor **(1)**

$$similarity(u) = \frac{\left(\sum_{v \in N(u)} JS(u, v)\right)}{|N(u)|} \quad \textbf{(1)}$$

where,

- u is the node of interest
- N(u), is the set of nodes in the neighborhood of node u,
- v, is a node and v∈N(u),
- C(u), C(v) = {t | t is category of restaurant node has commented on},
- |C(u)∩C(v)|, is the number of same categories of restaurants, that u and v have reviewed,
- |C(u)∪C(v)|, is the number of distinct categories of restaurants, that u and v have reviewed.
- JS(u, v), is

$$JS(u, v) = \sum_{v \in N(u)} \frac{|C(u) \cap C(v)|}{|C(u) \cup C(v)|}$$

Figure 4.4.1.1 shows the similarity of nodes in the actual network per degree. We notice that although the network is sparse, the similarity is quite high. This means that the first case is true, nodes tend to connect with similar others. But let's also check the similarity in the random network to make sure that this is not a matter of chance.

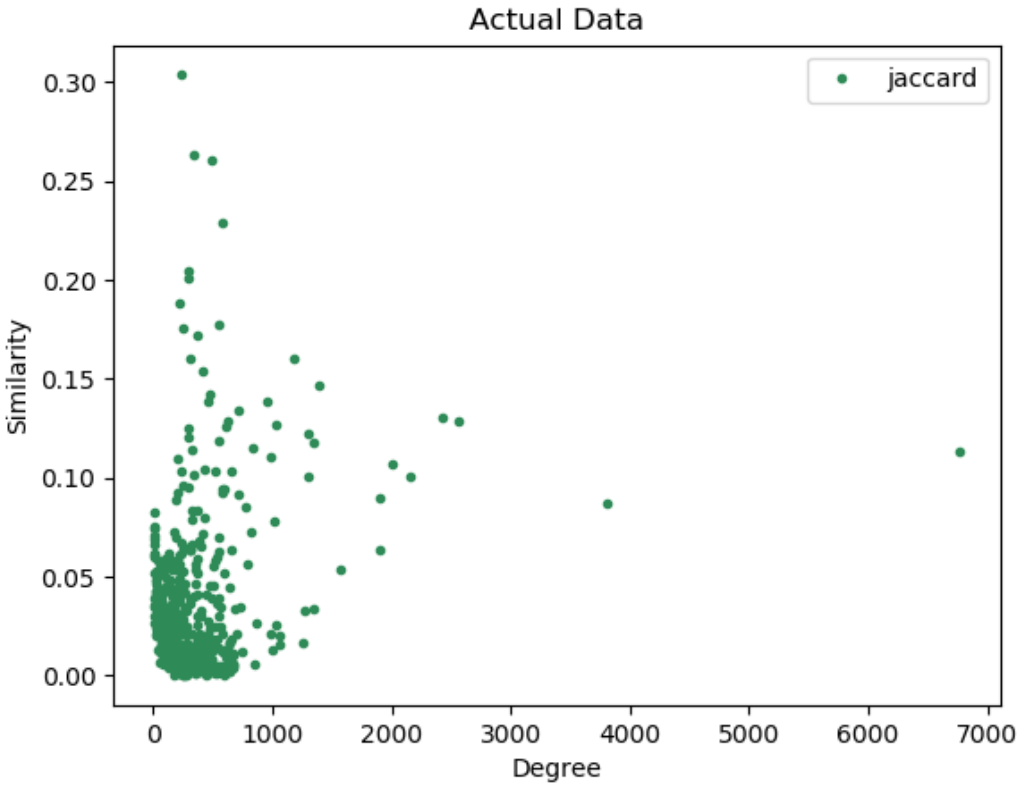


***Figure 4.4.1.1**: Similarity in actual network, per degree.*

To be fair, we don't only use one random network, but we use five random networks and we get the average of them. In particualar, we create five random networks as described in subsection 4.3. Then, we calculate similarity per degree for each network of them and define as similarity of a node, the average of the five networks' similarities **(2)**.

$$similarity\_in\_random(u) = \frac{\left(\sum_{1,\ldots,NoRN} similarity(u)\right)}{NoRN} \quad \textbf{(2)}$$

where,

- u is the node of interest
- NoNR, is the number of random networks, five in our case,
- similarity(u), is similarity as defined in equation **(1)**.

Figure 4.4.1.2 shows the similarity of nodes, by five random networks, per degree. We observe that the similarity is very low, although we have tried many different random graphs. It means that if nodes are connected with random nodes, they would not have much similarity to their neighborhood. So it is clear that nodes prefer to connect with similar ones.

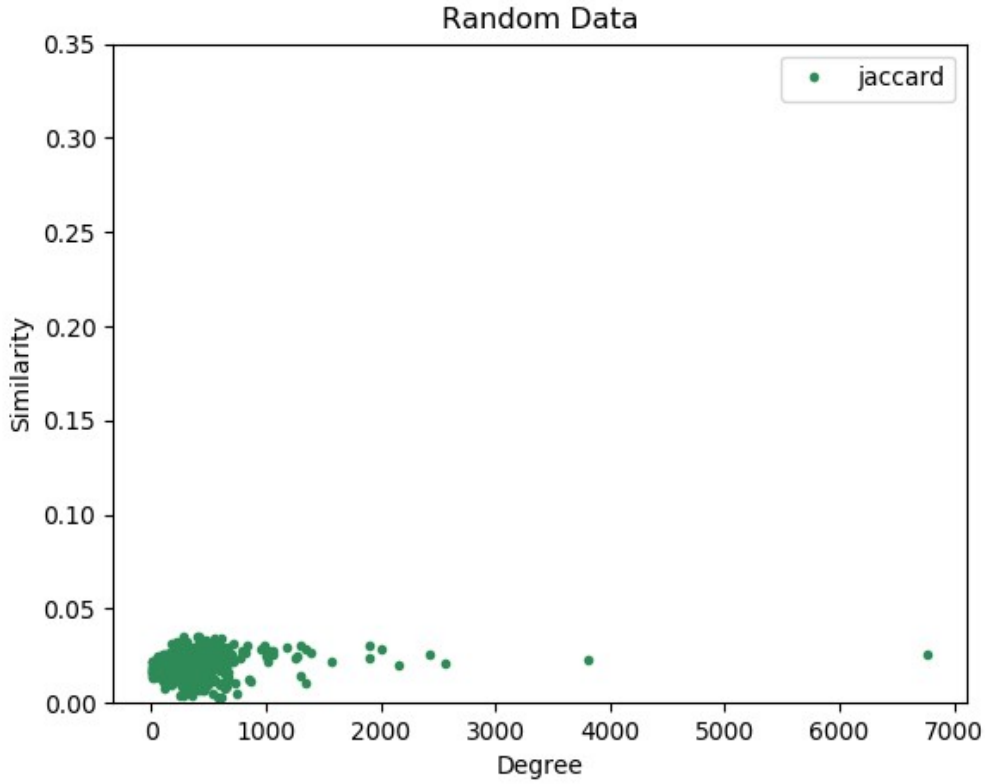
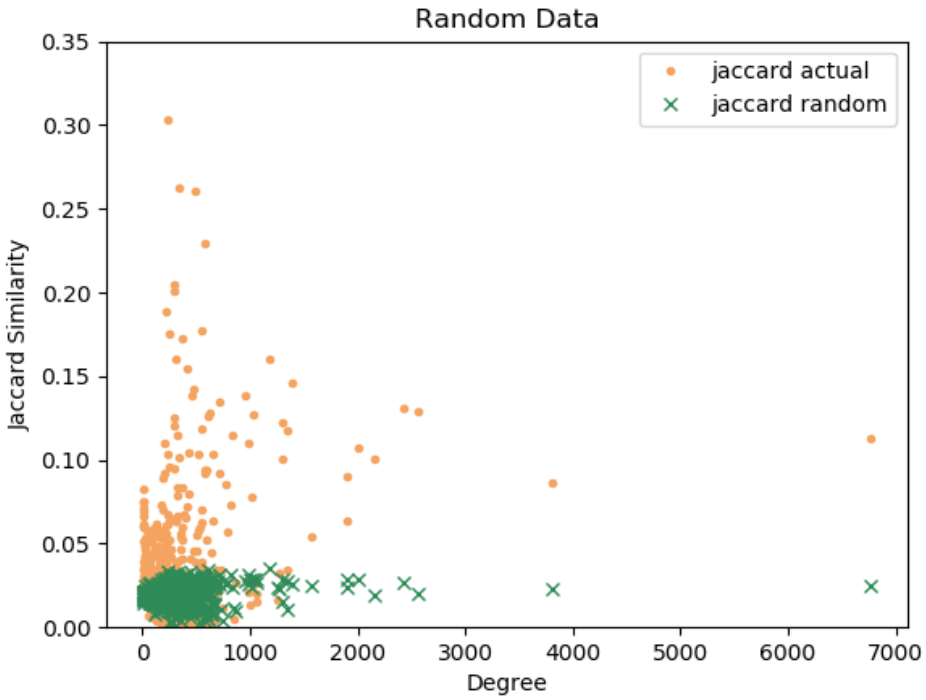


Figure 4.4.1.2*: Similarity in the random network per degree.*

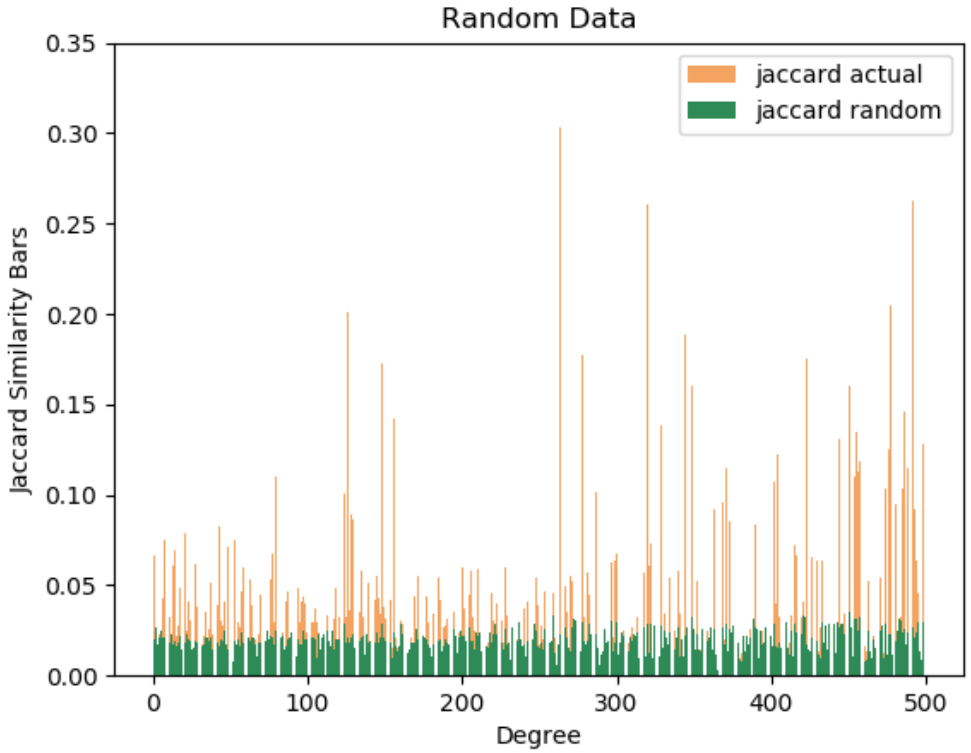
Figure 4.4.1.3 contains two plots. Both of them contain the same data, but they differ in the representation mode. They show, with orange color, the similarity of the nodes in the actual network and with green color, the similarity of the nodes in five random networks, per degree. We see that the similarity for nodes with same degree is higher in actual network, than in random networks.

In actual data networks, therefore, nodes tend to be similar to their neighbors. This allows them to live in a "small world" dominated by their own preferences and their neighbors' preferences. So they lose the feeling of what is generally in the "world".

So we conclude that the diversity in actual networks is low. It is up to us to examine what happens with link prediction and recommendation algorithms; if they reinforce this phenomenon by suggesting similar nodes in the network.



(a)



(b)

Figure 4.4.1.3*: Actual and random data.*

4.4.2 Link prediction results

In this subsection we use the actual network created in subsection 4.1.2 and the prediction algorithms, described in chapter 2, to consider if link prediction and recommendation algorithms suggest similar nodes for connection.
As we have found in the previous section, nodes in actual network tend to be similar with their neighbors. We want to check if link prediction and recommendation algorithms suggest also similar nodes. For this purpose we divide the network into two sets. The first is called training and the second test. The technique to do this described in section 2.4 of chapter 2. We run the algorithms using the training set and we get some results. Then, we check if there are common nodes in results and also in test sets. If there are such nodes, it means that the link prediction algorithm found nodes that are destination nodes; edges are created in the future. Next, we count the Jaccard similarity between nodes of care and destination nodes.
It should be noted that from all the algorithms that described in section 2, we use only methods based on neighborhoods, because our actual network has small diameter and large neighborhoods. So there would be no point in use of methods based on network paths.

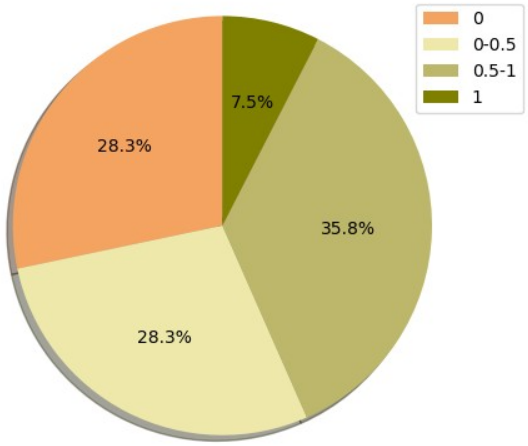


Figure 4.4.2.1: *Similarity in results of the algorithm uses Common Neighbors method.*

As we have seen, the actual network consists of 270,355 nodes and 385,001 edges. We study the connections of 4.314 nodes. The training set includes 315,136 edges and the test set includes 69,865 edges.
First, we apply the Common neighbors method to our training set. We get 65 correct future edges. ***Figure 4.4.2.1*** shows the similarity between the nodes of each new correct future edge. We notice that in most cases the similarity between the nodes is high.

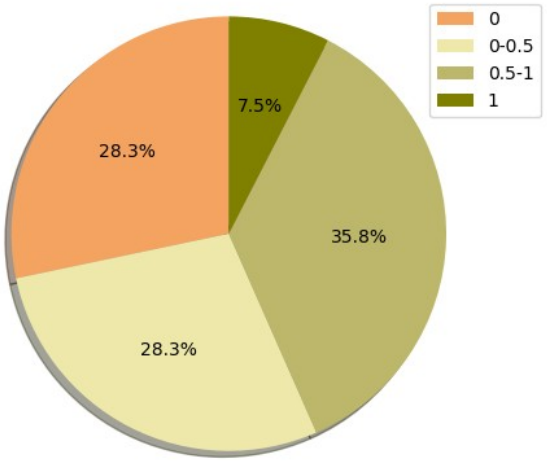


Figure 4.4.2.2*: Similarity in results of the algorithm uses Jaccard's coefficient method.*

Afterwards, we apply the Jaccard's coefficient method to our training set. We get also 65 correct future edges. ***Figure 4.4.2.2*** shows the similarity between the nodes of each new correct future edge. We observe that again the similarity between the nodes is high in the most cases.
We conclude that in both methods the similarity of the links proposed for connection is quite high. About 70% of the nodes are similar to the nodes to be linked. This means that link prediction works well. But in order to see if the methods tend generally to suggest similar nodes, we will look beyond the nodes that are actually connected in the future and the other results of the methods. The common neighbors method suggests 70,068 nodes. ***Figure 4.4.2.3*** shows the Common Neighbors similarity between nodes of care and suggested nodes.

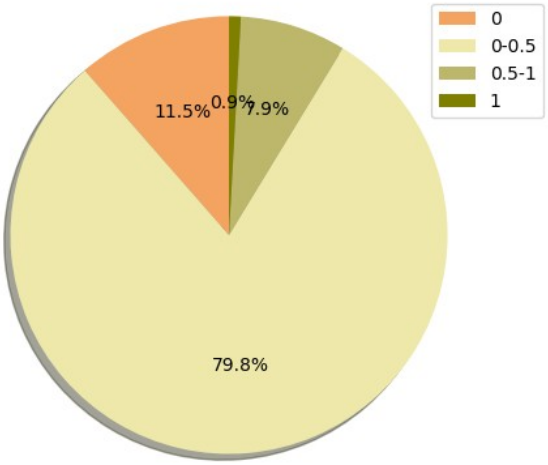


Figure 4.4.2.3*: Similarity in results of the algorithm uses Common Neighbors method for all suggested nodes.*

Similarly, the Jaccard's coefficient method suggests 70,068 nodes. ***Figure 4.4.2.4*** shows the Jaccard's coefficient similarity between nodes of care and suggested nodes.

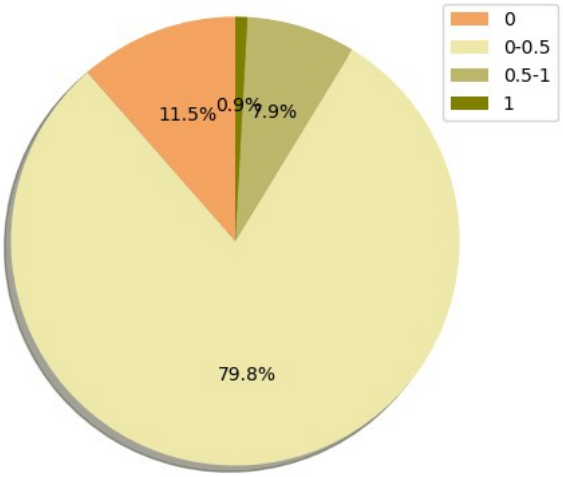


Figure 4.4.2.4*: Similarity in results of the algorithm uses Jaccard's coefficient method for all suggested nodes.*

We observe that about 90% of the suggested nodes are similar to nodes of care. We are sure now that, even in the actual data, link prediction and recommendation algorithms suggest similar nodes and this makes the diversity in the neighborhood even less.

Chapter 5

Conclusions and Future work

Summerizing, we studied the diversity in different networks. We created some synthetic networks taking into account the attributes of the nodes and we measured the mean similarity of them. We also ran some link prediction and link recommendation algorithms on them and from the results we saw, that the link prediction was good and that in general they predicted similar nodes to network nodes for connection. After we created an actual data network and some random networks and we measured the mean similarity of them. The similarity was high for the connected nodes in the actual network, so the diversity was low, while the similarity was low in the random networks. The link prediction and link recommendation algorithms also predicted similar nodes to nodes of the actual network for connection.

According to the above, we can now conclude, that the problem we study is an actual problem; nodes in social networks tend to connect with similar others and so the diversity in social networks is quite low. With regard to link prediction and link recommendation algorithms, they intensify this phenomenon with their predictions. We observed this in their results for the synthetic and the actual networks. It is also important to mention that the Supervised Random Walks with Restart algorithm, that takes into account the structure of the network and the attributes of the nodes, made much better link prediction than the algorithms that took into account only the structure of the network.

To close, we leave the way to combat the problem of low diversity for future work. One idea is to change link prediction and link recommendation algorithms, and especially the Supervised Random Walks algorithm, to suggest not only similar, but also not similar nodes for connection. So diversity will increase. One other idea for future work is to check whether this phenomenon also applies to other types of networks, other than social networks, such as collaboration networks for scientists or company emploees.

References

[1] Jure Leskovec, Anand Rajaraman, Jeffrey D. Ullman:
Mining of Massive Datasets, 2nd Ed. Cambridge University Press 2014, ISBN 978-1107077232

[2] David Liben-Nowell, Jon M. Kleinberg:
The link prediction problem for social networks. CIKM 2003: 556-559

[3] Lars Backstrom, Jure Leskovec:
Supervised random walks: predicting and recommending links in social networks. WSDM 2011: 635-644