DELFT UNIVERSITY OF TECHNOLOGY

MASTER THESIS

Link Prediction using Temporal Information in Multilayer Networks

Author:
Dionysios NIKOLOPOULOS

Supervisor:
Dr. Huijuan WANG

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Multimedia Computing Group
Department of Intelligent Systems

January 28, 2019

Declaration of Authorship

I

Living is no laughing matter:
you must live with great seriousness
like a squirrel, for example—
I mean without looking for something beyond and above living,
I mean living must be your whole occupation.
Living is no laughing matter:
you must take it seriously,
so much so and to such a degree
that, for example, your hands tied behind your back,
your back to the wall,
or else in a laboratory
in your white coat and safety glasses,
you can die for people—
even for people whose faces you've never seen,
even though you know living
is the most real, the most beautiful thing.

I mean, you must take living so seriously
that even at seventy, for example, you'll plant olive trees—
and not for your children, either,
but because although you fear death you don't believe it,
because living, I mean, weighs heavier.

Nazim Hikmet, On Living

DELFT UNIVERSITY OF TECHNOLOGY

Abstract

Faculty of Electrical Engineering, Applied Mathematics and Computer Science
Department of Intelligent Systems

Master of Science

Link Prediction using Temporal Information in Multilayer Networks

by Dionysios NIKOLOPOULOS

There is an increasing attention towards link prediction in complex networks both in physical and computer science communities. Particularly Online Social Networks (OSNs) are becoming the most popular platforms for information sharing, content creation and communication between users on the Internet. However, most of the research was done considering only a static snapshot of the network and without using relevant information from other types of activities.
In that direction, the present thesis proposes a novel method for link prediction using temporal information in Stack Overflow with the assistance of interactions from Github. The developed multilayer network enhanced with temporal interactions is aiming to improve the performance of the prediction compared to the traditional methods while the design choices intend to investigate the evolution of the network through time. In the end, the generalized framework could be used not only to make accurate link prediction that translate to human interactions over time, but also as a tool to characterize the behavior of the users in the targeted network.

Acknowledgements

The work presented here is the result of my research on the specific topic for my graduation project at Multimedia Computing Group in TU Delft. Under the supervision of Professor Huijuan Wang and PhD candidate Xiu-Xiu Zhan. I thank you both wholehartedly for the opportunity that I was given to tackle a problem far different to my background as a Telecommunications Engineer yet very interesting and meaningful.
Professor Wang, with your constant support and suggestions you pushed me to make a consistent story and yield meaningful results while I consider every meeting we had as a preparation for the final defence due to your targeted questions. Xiu-Xiu, your experience in the data I was given saved me a significant amount of time and your good advice helped me focus on the most important aspects that I had to consider.
It goes with out saying that I thank all the friends I made in and outside the University and opened a new window on how I perceive situations and made me more mature.
Last but not least, I thank my parents and my little sister for the endless amount of love that still in my 27 years of age I am not able to fully comprehend.

Contents

List of Figures

List of Tables

To Nikos, Despina and Maria…

Chapter 1

Introduction

1.1 Motivation

The well-received virtual environments increase the interactions among individuals providing more data to be used in social network analysis. In social networks, the members tend to interact in a highly dynamic manner; interactions appear and disappear along time, turning those networks into highly complex systems. Social Network Analysis (SNA) is the research domain that tries to deal with such complexity (Wasserman and Faust, 1994). The specific aim of the thesis is to investigate the prediction of links in social networks, that is the prediction of the most probable future interactions based on previous snapshots of the networks. This is a prominent problem of the SNA field called link prediction.
Link prediction deals with the occurrence of connections in a network. It consists of using the network structure up to a specific time so as to predict the appearance of links in the close future. The majority of previous work in link prediction is focused on proximity measures that assign scores to node pairs. For instance, new links can be predicted by sorting the proximity scores in decreasing order and pick the top ones. Link prediction (Lü and Zhou, 2011) appears in many applications, such as Recommender systems (Adomavicius and Tuzhilin, 2005), Viral Marketing, and Online Social Network Analysis (Dhote, Mishra, and Sharma, 2013). In more detail, Recommender Systems, such as applications for recommending books, CDs and other products, still remains an active research area because of the absence of practical applications that assist users to deal with the overload of information and provide personalized recommendations, services, and content to them. What is more, the highly dynamic nature of Social networks as they shrink and grow rather fast, thus evolving relationships among individuals or entities, makes link prediction a challenging task. Complex networks refer to networks with non-trivial features and patterns of connection that cannot be characterized regular or random. They can be considered as highly dynamic and evolving and with the current big data trend the analysis of the aforementioned network is a major challenge and active field of research for scientists (Lü et al., 2015).
A significant number of real-world systems can be modeled as networks that evolve over time. A network is a catalog the components of a system often called vertices or nodes and the direct interactions between them, called edges or links. These systems can refer to proteins, users and documents or social networks among others (Tabourier, Libert, and Lambiotte, 2016). The actors of the corresponding system are the nodes and edges represent any sort of interaction between them. In addition, another possible attribute of real-world networks is the heterogeneity that can be revealed in the form of multilayer networks; these networks consist of a layered structure with the same type of nodes but a different type of interaction in each layer.
A highlighted limitation observed in previous work is that the problem is approached in a static manner. Only the current state of the network is utilized and no temporal information is taken into account (Oyama, Hayashi, and Kashima, 2011). Particularly in social

networks, connections and actors tend to appear and disappear over time, justifying the system as dynamic and complex therefore this possibly useful information source should be exploited. Hence, in this thesis, a link prediction method is introduced that integrates temporal and multilayer networks using Machine Learning (ML). By exploiting the temporality that matches the evolution over time of social networks and the information from the multiple layers the performance of link prediction is improved.

1.2 Thesis Goal & Outline

The main task of the present work is to build and evaluate a temporal link prediction framework in multilayer networks. A novel method of link prediction is proposed that is applied in the Stack Overflow network. The goal is to exploit temporal information and additional information from Github network to further enhance the prediction. The approach is dictated by the need to uncover and understand how and what type of temporal information is critical and beneficial for the prediction task. The framework is compared to baseline static methods to assess its performance while machine learning methods were implemented to determine the finest outcome. The thesis is organized as it follows:

- Chapter 2 presents an overview of the basic network metrics and a short introduction to multilayer networks.
- Chapter 3 presents the link prediction methods that are used. First the baseline method and then the novel method introduced in the thesis. In the end, there is a discussion about the chosen performance metric that evaluates the method.
- In Chapter 4, the two networks under investigation are discussed with measurements and statistics that describe them. Then, the settings for link prediction in single and multilayer network are demonstrated.
- Finally in Chapter 5, the results of the method are laid out. The improvement of the performance is validated followed by a deeper investigation of the behavior of the studied networks. To conclude, the last section includes the contributions of the thesis to the link prediction domain.

Chapter 2

Structural measures of networks

2.1 Definition and examples of networks

The chapter focuses on the characterization of the structural properties of networks and it provides the formulation and definitions that will be used later. A network is a collection of connected objects that are named vertices or nodes and we customarily draw them as points. The connections between the nodes are called edges or links, and we usually draw them as lines between points. In other words, a network is a set of items (vertices or nodes) connected by edges or links.

There are many types of networks (Strogatz, 2001):

- Social and economic networks consist of a set of people or groups of people with some pattern of contacts or interactions between them. Facebook, friendship networks, labor markets, business relations between companies, are all examples of that type.
- Information networks refer to connections of "information objects". Some examples are: a network of citations between academic papers, semantic (how words or concepts link to each other), the World Wide Web (a network of web pages containing information with links from one page to other)
- Technological networks that are usually designed for distribution of a service. e.g., Internet (connections of administrative domains or routers), power grid, transportation networks (airline, road, rail, mail) as well as sensor networks and autonomous vehicles.
- A number of biological systems can also be represented as networks such as protein interaction network, food web, network of metabolic pathways.

2.2 Networks as graphs

It needs to be clarified that the terms networks and graphs are used with no distinction. In our case, the term graph is reserved for the abstract mathematical concept which usually refers to artificial formations of nodes and links. On the other hand, the term network is reserved for the graphs that represent real-world objects where nodes represent entities of the system and links represent the relationships among them. For that reason, it is clear that we will refer to the system of individuals and their interactions as a "social network" and not as a "social graph", nevertheless, they are identical. A graph $G(N, L)$, consists of a set of N nodes and L links that can be represented by an $N \times N$ adjacency matrix A where $A = [a_{u,v}]$. When there is an edge from node u to node v $a_{u,v} = 1$ and when there is no edge $a_{u,v} = 0$ (Albert and Barabási, 2002). The edge weight $a_{uv} > 0$ may also take non-binary values, describing the intensity of the interaction and the graph is called a weighted graph. We refer to a graph as a directed graph (or digraph) if $a_{uv} \neq a_{vu}$ and an undirected graph if $a_{uv} = a_{vu}$ for all $u, v \in N$.

2.2.1 Connectivity

Two nodes are adjacent when they are both incident to a common edge. A path in an undirected graph is a sequence of nodes such that a node v_i is adjacent to node v_{i+1} for $1 \leq i < n$. Such a path P is called a path of length $n-1$ from v_1 to v_n and if every two nodes are connected by a path then the graph is connected. If this is not the case, the components of a graph are the distinct maximally connected subgraphs.

2.2.2 Neighborhood and degree

The neighborhood of a node u is the set of nodes that u is connected to. What is more, for undirected graphs the degree of a node u is the number of edges that involve u (in other words the cardinality of his neighborhood). Regarding directed graphs degree is split in two: Node u's in-degree is $\sum_v a_{vu}$ and Node u's out-degree is $\sum_v a_{uv}$. While a small network can be depicted directly by its graph, larger networks are usually more difficult to describe. Thus, a set of quantitative performance measures and statistics is used to compare and characterize networks with a focus on undirected graphs.

2.2.3 Diameter and average path length

Let $H(u,v)$ denote the length of the shortest path between node u and v. The largest distance between any two nodes in the network is called the diameter of a network:

$$H_{max} = \max_{u,v}\{H(u,v)\} \tag{2.1}$$

the average distance between any two nodes in the network is the average path length:

$$E[H] = \frac{\sum_{u \geq v} H(u,v)}{\frac{n(n-1)}{2}} \tag{2.2}$$

also:

$$H_{max} = E[H]$$

2.2.4 Clustering

Clustering is the tendency of nodes to cluster together and in order to measure the degree of that behavior the clustering coefficient is introduced. Clustering can be represented by the global clustering coefficient $C(G)$, given by

$$C(G) = 3 \times \frac{\text{number of triangles in the network}}{\text{number of connected triples of nodes}} \tag{2.3}$$

where a "connected triple" refers to a node with edges to pair of nodes. As it can be seen:

$$0 \leq C(G) \leq 1 \tag{2.4}$$

C(G) measures the fraction of triples that have their third edge connected and complete the triangle. It is also referred to as network transitivity while another measure of clustering is defined on an individual node basis: The local clustering for a node u is:

$$C_u = \frac{\text{number of triangles connected to node } u}{\text{number of triples centered at } u} \tag{2.5}$$

The average clustering coefficient is

$$C = \frac{1}{n}\sum_{u} C_u \tag{2.6}$$

2.2.5 Centrality

Centrality measures the importance of a node's position in the network and consists one of the fundamental groups of network metrics. There are different measures of centrality and three are mentioned below:

- *Degree centrality*: It is the degree of the node u.
- *Closeness centrality*: Measures how close a given node is to any other: for a node u, it is expressed as:
$$\frac{n-1}{\sum_{u \neq v} H(u,v)} \tag{2.7}$$
where $H(u,v)$ is the path length between u and v.
- *Betweenness centrality*: It measures the extent to which a node lies on paths between other nodes. Nodes with high betweenness could have substantial influence within a network as they they control the information passing between others. The formula of betweenness is given as follows:
$$B(u) = \sum_{s \neq u \neq t} \frac{\sigma_{st}(u)}{\sigma_{st}} \tag{2.8}$$
in the equation above σ_{st} is the total number of shortest paths from node s to node t and $\sigma_{st}(v)$ is the number of those paths that pass through v.

2.3 Weighted and Multilayer Networks

A lot of work has been done in investigating the dynamics and structure of complex systems. A large number of these could be represented as a network whose nodes serve as the different fundamental units of the system, and whose links represent the relationships/interactions among the units (Albert and Barabási, 2002). A common approach to network description consists of the aggregation of all the links observed between a certain set of basic units. Nevertheless, that aggregation could discard significant information about the function and structure of the original system; since elementary constituents of a system might be connected through many kinds of relationships that differ in meaning and relevance (Kurant and Thiran, 2006). For example, the same set of entities in a social system could be connected through collaboration, commercial kinship, friendship or communication relationships, while in complex transportation systems - a typical situation in large metropolitan areas - locations might be reached in different ways; namely suburban rail, bus, underground networks just to name some of them. For these systems, each type of interaction has been assigned a given importance, distance, relevance, cost or meaning, thus treating all the links as being equivalent, results into losing an important amount of information. A finer description of such systems is in terms of multilayer networks. Multilayer networks refer to networks where each node appears in a set of different layers and each layer describes a different type of edges.

For example, a set of nodes represents individuals that interact with each other. However, there are many ways of interactions between the same group of people: a single person has

social, professional, personal circles, but could also have multiple accounts in online social systems (Twitter, Facebook, etc.). In many cases, we deal with that diversity by projecting all those layers into a single layer but not all processes can be transformed successfully on that simplified aggregated network.
Another relevant example could be the evolution of disease on a particular network. In many situations, a large number of pathogens coexist within the same host population and interact with each other. For example, there are systems of competing pathogens (e.g., seasonal influenza or HIV and Tuberculosis), i.e., two pathogens each of which impedes or enhances the spreading of the other. For these scenarios, the use of multilayer networks is paramount. In single layer networks that are unweighted and undirected, the degree of a node gives the number of nodes that are adjacent to it (that is the number of its immediate neighbors). The notion of degree can be generalized for directed networks - networks that their edges have a certain direction - to obtain in-degree and out-degree (Kivela et al., 2014). In weighted networks where each edge can be assigned a weight, thus not all edges bear equal importance, weighted degree or strength is introduced. The weighted degree is given by the sum of the weights of all edges that are incident to a node.
There is a number of ways to generalize the notions of degree and neighborhood for multilayer networks, however, one obtains the usual and aforementioned definitions if he considers only a single intra-layer network at each time.
The fundamental way to generalize degree and neighborhood for multilayer networks is to use network aggregation. Then, the degree is defined as the number of edges (any type) that are incident to a node and by following the same logic, the neighborhood is the set of nodes that can be reached from a specific node by following any of those edges. Another way is to place a threshold in an aggregated network such that two nodes are considered to be adjacent if and only if the number of edges that connect them in a multilayer network is larger than some value (Lytras et al., 2010). In the end, the traditional way to examine systems that are multilayer is to construct a single-layered network by aggregating data from the different layers of the multilayer networks. Then the resulting single layer network is studied (De Domenico et al., 2013). Hence, standard network techniques can be applied and it is sometimes a desirable way to help alleviate issues with data that are noisy.

2.4 Temporal Networks

A great variety of systems can be modeled as graphs of nodes coupled by links. The structure of the network that describes how the graph is wired, helps us understand the behavior of those systems. However, in many cases, the links are not continuously active. For example, in communication networks such as email or phone calls, links represent sequences of practically instantaneous contacts while there are also cases where links are active for non-negligible time periods. The temporal structure of link activations can affect the dynamics of systems such as information diffusion over the aforementioned e-mail network.
The temporal networks can be roughly divided into two classes. In one case, the system can be represented by a contact sequence which is a set of C contacts, namely triples (u, v, t) where $u, v \in V$ and t denotes time. Some systems suitable to be represented as a contact sequence include physical proximity data where the duration of the contact is less important and communication data (sets of phone calls, e-mails). With this representation, it is tempting to think of the temporal network structure as a static network structure that evolves over time.
The second class refers to links that are active over a set of intervals $T_e = (t_1, t_1'), ..., (t_n, t_n')$, where the parentheses mark the activity periods the unprimed times indicate the beginning of the interval and the primed quantities mark the end. The static graph with a link between i and j if and only if there is a contact between i and j is called the time-aggregated

graph. Some examples of systems that can be modeled in that way include infrastructural systems like the Internet, proximity networks (where a contact represents that two entities have been close to each other for some time) and seasonal food webs where a time interval can represent that one species is the main food source of another at some time of the year (Holme and Saramäki, 2012). It is useful to define an index function of whether a pair of nodes is connected at a given time:

$$a(u,v,t) = \begin{cases} 1, & \text{if } u \text{ and } v \text{ are connected at time t} \\ 0, & otherwise \end{cases} \tag{2.9}$$

It will be shown in Chapter 3 that the created temporal networks belong to the latter class since links are considered active for specific time windows (weeks).

2.5 Social Networks

Since the networks that are analyzed in the thesis are social, it seems appropriate to proceed to a further discussion about them.
Social networks are defined by structures where nodes represent people or other entities embedded in a social context and edges could represent collaboration, interaction or influence between the nodes (Liben-Nowell and Kleinberg, 2007). These networks have several notable characteristics, such as the small world property (Watts and Strogatz, 1998), power law degree distribution (Barabási and Albert, 1999), and the community structure (clustering effect) (Girvan and Newman, 2002).
The *small world effect* indicates that the average distance in the network is significantly small with respect to the size of the network. That means that every pair of nodes can be connected through a relatively short path in the network. Apart from the famous experiments of Stanley Milgramm (Travers and Milgram, 1967), in 2014, Facebook using the entire network of active users (721 million users, 69 billion friendship links) performed their first world scale social network graph-distance computation, It was found that the average distance was 4.74, corresponding to 3.74 "degrees of separation" (Backstrom et al., 2012).
The *scale-free effect* refers to the phenomenon only a few nodes have a large number of links. Nodes with high degree are called hubs and dominate the network operation. Scale-free effect illustrates that degree distribution is strongly uneven in large-scale networks. As a matter of fact, world-wide web follows the aforementioned distribution.
Clustering effect introduces the appearance of small groups in social networks. There is a circle of friends, acquaintances and each member knows each other. It can be further described by the concept of triadic closure: there are many fully connected subgraphs in a social network.

2.6 Data and Settings

In this section, the data used on this thesis are shortly discussed. Firstly, Stack Overflow network is used, a popular online community, that has clearly a social nature with users interacting with timestamped actions. Then, for the realization of the multilayer network, matched users from Github network are integrated into the aforementioned setting.

2.6.1 Stack Overflow

Stack Overflow is a website created in 2008 by Joel Spolsky and Jeff Atwood. It was developed to be an efficient and friendlier alternative to earlier question and answer sites such as Experts-Exchange. The website serves as a platform for users to ask and answer questions

on a wide range of topics in computer programming and through membership and active participation, to edit or vote questions and answers up or down.
A study in 2013 (Asaduzzaman et al., 2013) found that 77% of users ask only one question, 65% answer only once, and only 8% of users answer more than 5 questions. From 2011 as a starting year, 92% of the questions were answered, in a median time of 11 minutes. Nowadays, the software deletes questions that meet certain criteria automatically, e.g. having no answers in a specific amount of time.
In 2016, 1.5 million posts were deleted, of which about 8% were deleted by moderators.

2.6.2 GitHub

GitHub is a web-based Git repository and Internet hosting service mainly used for code. It provides all of Git's source code management functionality and distributed version control and more than 28 million people use GitHub to discover, fork, and contribute to over 85 million projects.

Chapter 3

Link Prediction Methods

3.1 Introduction

This chapter presents the methods used to perform link prediction in the studied networks. After the problem definition, a method from the literature that provides the basic framework is discussed (Soares and Prudencio, 2013). Then, the novel method is introduced followed by a discussion about the selected performance metric (Area Under Curve - AUC).

3.2 Problem definition

A common definition of the link prediction problem is the following: "Given a snapshot of a network at time t we attempt to correctly predict the links that will appear in a given future time". To extend the problem we can rephrase: "Given link data for times 1 through T, can we predict the links at time T+1? If our data has an underlying periodic structure, or can we predict out even further? i.e., links at time T+2, T+3, etc.
More formally, in a network $G = (N; \ L)$, where L is the set of links N is the set of nodes, the link prediction task is to predict whether there is or there will be a link $e(u; \ v)$ between a pair of nodes u and v. Usually, the link prediction problem falls into two groups:

1. Link prediction is utilized to predict future collaboration or friendship thus it is targeted to predict links that will be added to an observed network in a future time. That setting can be useful for exploring underlying mechanisms that govern network evolution (Sharma and Singh, 2015).

2. Link prediction is used to discover lost or hidden links of a network (Yang and Zhang, 2016). As an example, it can be deployed to infer unobserved protein-protein interactions.

However, predicting links is a particularly hard problem especially in social networks where sparsity is common property.
An illustrative example is the DBLP dataset (Al Hasan et al., 2006), where in the year 2000, the ratio of actual to possible link was around as $2 * 10^{-5}$. Thus, only a small group of nodes interacted with each other, whereas the majority was inactive. Hence, in a uniformly sampled dataset with one million node pairs, we can expect only 20 actual links. Even worse, the aforesaid ratio is slowly decreasing.
From 1995 to 2004 the number of authors in DBLP increased from 22 thousand to 286 thousand, meaning that the actual collaborations increased only by a factor of 21 while the possible collaborations increased by a factor of 169.
In the present case, for the Stack Overflow network, the ratio is even smaller. Its value is about $9 * 10^{-6}$, highlighting the class imbalance problem that will be discussed later. An attempt to summarize in a systematic way how the link prediction scheme works in the majority of the cases is shown below:

1. Input is the current state of the network, usually static.
2. Proximity measures - that are presented in Section 3.3 - are deployed and assign a score to each node pair under test.
3. An unsupervised or supervised algorithm is deployed to classify the nodes pairs as it is explained below.

Regarding the last step, an unsupervised method could be sorting the proximity scores in a decreasing order and pick the highest ranked ones as the predicted future links, while a supervised method could have a set of different proximity measures as features (e.g. *Common Neighbors*, *Preferential Attachment*, *Jaccard Coefficient*) to train a predictor for binary classification. The binary result of the latter will decide if the node pair will be present or not in the future.
As stated, for link prediction, the chosen features should represent some form of proximity between the pair of vertices and in the majority of the cases, features are extracted from the graph topology. It should be mentioned that the attributes of vertices and edges can also be features for some application domains (Millen and Feinberg, 2006). Features that are based on graph topology are the most natural for link prediction and typically, they compute the similarity based on the ensembles of paths between node pairs or based on the neighborhood of the nodes. The main advantage of them is that they are applicable for graphs in any domain meaning that no domain knowledge is necessary to compute their values. Nevertheless, for large networks, some of these could be computationally expensive. In any case, the most popular topological measures that all under two categories: *Node neighborhood based* and *Path based*.

3.3 Baseline Methods

Roughly all topological measures are categorized into neighborhood-based or path-based measures (Al Hasan and Zaki, 2011). The neighborhood-based measures take into account the immediate neighbors of the nodes and generally consider that if the sets of neighbors of two nodes have a large overlap, then they are more likely to form a link. On the other hand, path-based measures define proximity considering the paths between the nodes under test (Xiang, 2008).
The basic concept is that two nodes are more likely to form a link if there are short paths between them. These measures range from the elementary path-distance measure to more complex definitions that take into account ensembles of different paths, such as the Katz measure (Katz, 1953). Nevertheless, the neighborhood-based methods are more common, due to both their great performance and computational efficiency observed in experiments (Chen, Huan, and Ma, 2006) (Murata and Moriyasu, 2008).
Some basic neighborhood-based measures are Preferential Attachment, Common Neighbors, Adamic and Adar, and Jaccard's coefficient (Salton and McGill, 1986). *PA* measure speculates that the probability of an upcoming link between two nodes is proportional to their degree. It is justified by the notion that in some social networks, namely finance, "rich" nodes get "richer". It is defined as:

$$PA(u,v) = |\Gamma(u)| \times |\Gamma(v)| \tag{3.1}$$

where $\Gamma(u)$ refers to the neighbors of node u and $|\Gamma(u)|$ is the total number of them. *PA* does not require neighborhood information, therefore, has low computational complexity.
CN measure assumes that the bigger the number of neighbors two nodes share, the higher is the probability to form a link in the future ((Newman, 2001)). The measure is defined as:

$$CN(u,v) = |\Gamma(u) \cap \Gamma(v)| \tag{3.2}$$

The AA measure refines the CN by favoring the node pairs that the neighbors in common possess fewer connections. More formally:

$$AA(u,v) = \sum_{z \in \Gamma(u) \cap \Gamma(v)} \frac{1}{log|\Gamma(z)|} \tag{3.3}$$

Lastly, *JC* assigns higher proximity scores for pairs of nodes which share a higher proportion of common neighbors relative to their total number of neighbors.

$$JC(u,v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|} \tag{3.4}$$

All of the measures above were applied to the network under investigation in order to compare their performance with the method introduced on the thesis. The reason for the particular selection of the topological measure above is their massive adaptation. They consist the baseline, the point of reference in the majority of link prediction problem and are treated as such in the present thesis. The proposed method yielded better results meaning that it assigns scores to node pairs in a better manner. The way these scores are assigned and the performance metric that is used will be presented in the following chapter.

3.4 Event-based link prediction

3.4.1 Introduction

Event-based link prediction will be introduced by an example; we can make the assumption that researchers make collaborations when they meet (in a conference, forum etc.) or when they collaborated with the same people, a predictor can be established based on the network structure consisting of nodes-researchers and links-collaborations. Hence, in this case, the problem of link prediction refers to predicting future partnerships. However, rather frequently, the graph structure is very sparse and does not allow the usage of classical graph metrics. In that case, an event-based approach can be beneficial. If one was aware to what extent each researcher is an expert in each field, he could possibly use this knowledge to find researchers with similar expertise provide suggestions. As an event-based approach, we can consider the common journal where researchers published their works or common conferences where they presented. Those could contribute to identify the potential future collaborations with more precision.
Social networks are highly dynamic as interactions appear and disappear over time. The evolution of these networks provides valuable information about how connections tend to be formed. Based on that and previous work on that domain, a proximity measure is proposed that takes into account the temporal structure of the network. Specifically, it introduces temporal events between every two nodes of the network. The goal is to integrate the notion that the strength of a connection between two users is straightforwardly associated with the frequency they interact with each other (Homans, 1958) with Newman's idea that the larger the number of common neighbors between two users, the higher is their probability to be connected in a future time (Newman, 2001).
The methodology of exploiting the temporality of a network is based on previous research and is presented in the next subsection.

3.4.2 Temporal Events

A temporal event is defined as a distinct activity between two nodes from a frame to its subsequent (Soares and Prudencio, 2013). It can be seen as an action that shifts the pair of nodes from a state to another (connected or non-connected). Events can be grouped into one of three mutually exclusive types that are represented with the help of contact sequences: In order to uncover the temporal information, a temporal structure is formed from the timestamped network under investigation. In general, the network is split into consecutive time-frames whose size can change and in the present case is equal to one week.

- Conservative

  A conservative event occurs when an interaction between two nodes is maintained when the network evolves, that is, when $a(i,j,T)=a(i,j,T+1)$. Moreover, a reward is defined in order to take into account a conservative event during the corresponding transition. The reward quantifies the effect of a conservative event in the link prediction scheme. Formally:

  $$C(u,v,k)=\begin{cases} c, & if \quad a(u,v,k)=1, \quad a(u,v,k-1)=1 \\ 0, & otherwise \end{cases} \tag{3.5}$$

  Where E are the frames and k the index of the frame

- Innovative event refers to the creation of a link. In a formulated way:

  $$I(u,v,k)=\begin{cases} i, & if \quad a(u,v,k)=0, \quad a(u,v,k-1)=1 \\ 0, & otherwise \end{cases} \tag{3.6}$$

  Again, i represents the reward for each Innovative event.

- Regressive event is the exact opposite of an Innovative event. It signifies the removal of an existing link in the next time step. Conceptually, that behavior reveals a weakening in the relationship of the node pairs, thus, it should be assigned a negative-valued reward r.

  $$R(u,v,k)=\begin{cases} r, & if \quad a(u,v,k)=1, \quad a(u,v,k-1)=0 \\ 0, & otherwise \end{cases} \tag{3.7}$$

The values of the rewards would be thoroughly discussed in the next chapter as they bear great significance for the performance of link prediction.

3.4.3 Event-based Score

Since temporal events were introduced, the next step is to combine and integrate them in a way that determines the similarity of the node pair accurately. Most approaches for link prediction assign scores to pairs of nodes by deploying a chosen proximity measure, trying to determine the similarity of those nodes and consequently, how likely is for a connection between them to happen in a close future. The proposed measure takes into account events that pertain to the neighborhood of each examined node pair by breaking down an event to two categories: Primary and Secondary events. That means that for each node pair under investigation a possible event belongs to one of the two aforementioned categories. In order to do that, it should be mentioned that the neighborhood of a node in the temporal network consists of the accumulated interactions of it over time. More specifically, the neighborhood of a node at time T is the set of nodes that the node interacted in the past

until time T. This information is used to perform link prediction for time T+1.
For a node pair, the temporal events that appear over time strictly between those two nodes are named Primary events while temporal events that occur in the common neighborhood of the node pair are considered Secondary events. These are the essential parts of the proximity score that is assigned to a given pair of nodes$(u, \; v)$ and is neatly summarized as:

$$Score(u,v) = \sum_{k=2}^{n} P(u,v,k) + \alpha S(u,v,k) \tag{3.8}$$

$$P(u,v,k) = C(u,v,k) + I(u,v,k) + R(u,v,k) \tag{3.9}$$

$$S(u,v,k) = \sum_{y \in \Gamma(u) \cap \Gamma(v)} P(u,y,k) + P(y,v,k) \tag{3.10}$$

Equation (3.8) states that the proximity score is the linear summation of all Primary and Secondary events over time. However, Secondary events are multiplied by an amortization factor that indicates how powerful is the effect of Secondary events on a possible tie between the nodes of the node pair. Equation (3.9) clarifies that the value of the Primary events is again a linear summation of all the temporal events that took place during the evolution of the network. Equation (3.10) is expanding the notion of (3.9) to the Secondary events; the value of such events is the summation of all Primary events between the node pair's common neighborhood and each node of the node pair. The upcoming figure offers an example of the proximity measure above.

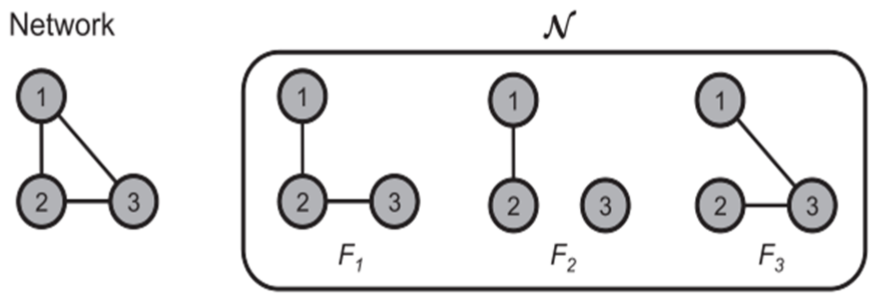


FIGURE 3.1: Temporal Decomposition of a Network (Soares and Prudencio, 2013)

To calculate the score of node pair (1,3), we look into the events that were created in the 3 time-frames.

Primary Events
From F1 to F2 that are shown in the figure, there are no events and from F2 to F3 there is an Innovative event with the corresponding reward.
Secondary Events
These events correspond to node pairs (1,2) and (2,3):
Node pair (1,2): From F1 to F2 a conservative event c takes place and for F2-F3 transition a Regressive event with reward r.
Node pair (2,3): From F1 to F2 a Regressive event r takes place and for F2-F3 transition an Innovative event with reward i. Therefore:

$$Score(1,3) = i + \alpha * (c + 2 * r + i) \tag{3.11}$$

Subsequently, these scores were used by an unsupervised algorithm in the previous work; every node pair is ranked based on its proximity score and the top-ranked ones are selected as the predicted new links. The novel method presented in the thesis introduced a supervised algorithm as it will be shown in Section 3.5.
Eventually, proximity score becomes a function of the 3 rewards that are described above and the amortization factor α, hence the performance of the algorithm is dependent on them. That being said, the goal is to find the optimal rewards for each temporal event to achieve the best performance. In the previous work, rewards were calculated by using a brute force method. The present thesis integrated a Machine Learning approach that attempted to find the global best rewards of each event.

3.4.4 Brute Force

As stated above, (Soares and Prudencio, 2013) used a brute force scheme. The reward of the innovative event was normalized to 1 and regressive and conservative events varied between 2 fixed values (0 - 4 with 0.5 step). The same approach was followed for the acquisition of the best α.
Those facts introduce certain considerations that need to be addressed. A brute force approach apart from its time inefficiency (e.g. it usually requires large parametric sweeps) is also not optimal. Since only a pre-defined set of values is tested there is no proof that the best reward derived from that set is in fact, the best.
Moreover, a more general remark would be the constraint of the reward's value regarding Primary and Secondary events. Both events have the same reward value for the corresponding temporal event and the only thing that differentiates them is the amortization factor. Thus, tweaking the basic functions and detaching the rewards of Primary and Secondary events could provide better insight into the network mechanisms.

3.5 Event Based Link Prediction with Machine Learning

For the reasons above, a Machine Learning (ML) scheme is proposed here to derive the best rewards for the temporal events (Pedregosa et al., 2011). This is the novel method proposed by the thesis and is also extended to multilayer networks. ML will provide a systematic way of picking the best rewards for each occasion but reformatting of the basic equations is required.

6 rewards: 2 groups of (c,r,i) one for Primary and one for Secondary events.

Remove amortization factor: By adding separate rewards for Secondary events amortization factor is useless and apart from that its removal causes the scoring equation to become linear.

Select a suitable ML algorithm: Since link prediction can be characterized as a binary classification problem Support Vector Machines (SVM) was chosen as a supervised learning method for classification. The modification of the equations is shown below:

$$Score(u,v) = \sum_{k=2}^{n} P(u,v,k) + S(u,v,k) \tag{3.12}$$

$$P_{primary}(u,v,k) = C_p(u,v,k) + I_p(u,v,k) + R_p(u,v,k) \tag{3.13}$$

$$P_{secondary}(u,v,k) = C_s(u,v,k) + I_s(u,v,k) + R_s(u,v,k) \tag{3.14}$$

$$S(u,v,k) = \sum_{y \in \Gamma(u) \cap \Gamma(v)} P_{primary}(u,y,k) + P_{secondary}(y,v,k) \tag{3.15}$$

Support Vector Machines

Although the scope of the present thesis was not meant to include a detailed explanation of the SVM algorithm, it is crucial to understand how is linked with the calculation of the reward's value.

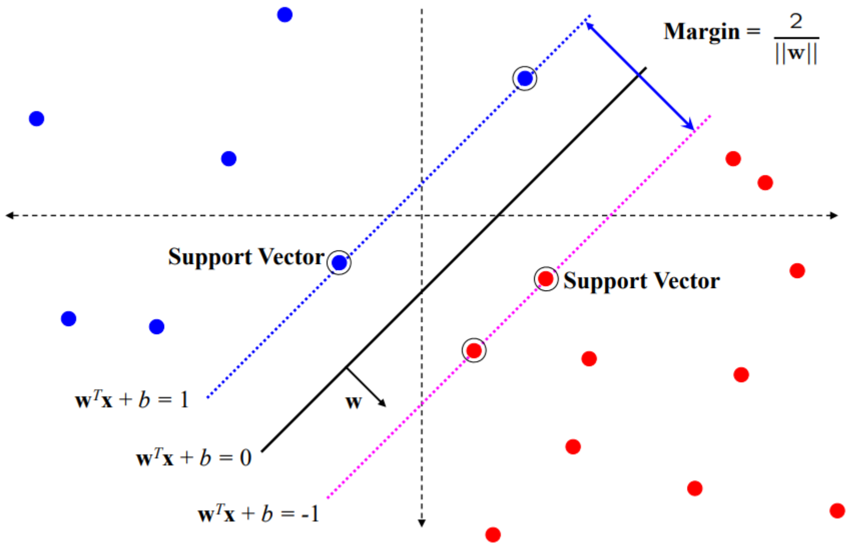

Margin = $\frac{2}{||\mathbf{w}||}$
Support Vector
Support Vector
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}$
$\mathbf{w}^T\mathbf{x} + b = 0$
$\mathbf{w}^T\mathbf{x} + b = -1$

FIGURE 3.2: SVM Classification (Sayad, 2017)

The goal of an SVM method is to find the hyperplane that gives the largest minimum distance to the training examples (Hackeling, 2014). The instances that are closer to the decision boundary are called support vectors and if removed, they would alter the position of the boundary. Hence, they can be considered the critical elements of the data set. Figure 5 shows a number of instances that belong to two classes and SVM attempts to separate them by maximizing the margin. Due to its two-dimensional character, two features are assigned to each instance and their values correspond to the position they reside in the two-dimensional plane. The mathematical representation of the optimization that takes place in SVM is:

$$min||\mathbf{w}||^2_{wR^d} + C\sum_{i}^{n} max(0, 1 - y_i f(\mathbf{x}_i)) \tag{3.16}$$

$$f(\mathbf{x}_i) = \mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \tag{3.17}$$

In (3.16) the first part maximizes the margin as shown in the figure while the second part is a misclassification penalty. The margin above can be considered as a soft margin; $\mathbf{y}_i$ is the class of each instance (1 or -1) and if the classification is correct the product $y_i f(\mathbf{x}_i)$ is positive thus zero penalty. Otherwise, a penalty is introduced proportional to the product. In the present case, a six-dimensional space is constructed where the six different kinds of temporal events are the features of the algorithm. Once more, that is a binary classification problem but the position of each instance in the six-dimensional plane is yet to be found by optimizing the equations above. The result of the algorithm will provide the 6 coefficients of the decision boundary that separate the two classes optimally. Those six values are the values of the rewards. In general, the advantages of support vector machines are:

- Memory efficiency due to the use of a subset of training points in the decision function
- Versatility as different kernel functions can be specified for the decision function; decision function can be linear, polynomial, sigmoid etc. However, only the linear kernel was used to derive the weights of the features that in our case are the rewards of the temporal events.
- Effective in high dimensional spaces

For that purpose, the Linear Support Vector Classifier from Scikit-learn machine learning library was selected. Its ability to tackle class imbalance and simple extraction of the desirable coefficients were the reasons for that choice. It should be noted that more classification algorithms were used such as Logistic Regression and Stochastic Gradient Descent. They both performed worse and a possible explanation could be that Logistic Regression calculates different loss (logistic loss) from SVC (hinge loss). Logistic loss diverges faster than hinge loss and it will be more sensitive to outliers and additionally, does not go to zero even if the point is classified sufficiently and confidently. This could lead to minor degradation in accuracy.

3.5.1 Tackling Class Imbalance in Classification

Although class imbalance was discussed in the introduction it is essential to observe the implications risen in classification. Most ML algorithms give optimal results when the number of instances of each class is roughly equal. When the number of instances of one class far exceeds the other, problems arise. The evaluation criteria are a key factor in determining the classification performance and perform the classifier modeling. In binary classification, the confusion matrix (shown in Table 3.1) records the results of correctly and falsely recognized examples of each class.

TABLE 3.1: Confusion Matrix

	Positive prediction	Negative prediction
Positive class	True Positive (TP)	False Negative (FN)
Negative class	False Positive (FP)	True Negative (TN)

Customarily, accuracy (3.18), has been the most popular empirical measure. Nevertheless, in an imbalanced framework, accuracy is no longer suitable, since it does not distinguish between the number of correctly classified instances of different classes. Thus, it can draw false conclusions, i.e., in a dataset with class imbalance ratio equal to 9, a classifier that places all instances in the majority class and misclassifying all items that belong to the minority will have an accuracy of 90%.

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \tag{3.18}$$

In imbalanced domains, specific metrics should be deployed to evaluate the performance of the classifiers by taking account the class distribution. Those stem from Table 3.1 and are presented below:

- True positive rate:$TP_{rate} = \frac{TP}{TP+TN}$, percentage of positive instances correctly classified
- True negative rate: $TN_{rate} = \frac{TN}{FP+TN}$, percentage of negative instances correctly classified

- False positive rate: $FP_{rate} = \frac{FP}{FP+TN}$, percentage of negative misclassified instances
- False negative rate: $FN_{rate} = \frac{FN}{TP+FN}$, percentage of positive misclassified instances

ROC - AUC & PR - AUC

A well-known approach that combines those measures to provide good results and legitimate evaluation is the use of the Receiver Operating Characteristic (ROC) (Fawcett, 2004). The ROC curve is constructed by plotting the TP_{rate} against the FP_{rate} at various threshold settings. This graphic allows the visualization of the trade-off between the benefits (TP_{rate}) and costs (FP_{rate}), as it demonstrates that classifiers cannot increase the number of true positives without increasing the false positives at the same time. The Area Under the ROC Curve (ROC-AUC) depicts the probability of correctly identifying which observation is signal plus noise and which one is just noise. Moreover, it yields a single value for evaluating a classifier's performance; it provides the opportunity to decide which model is better on average.

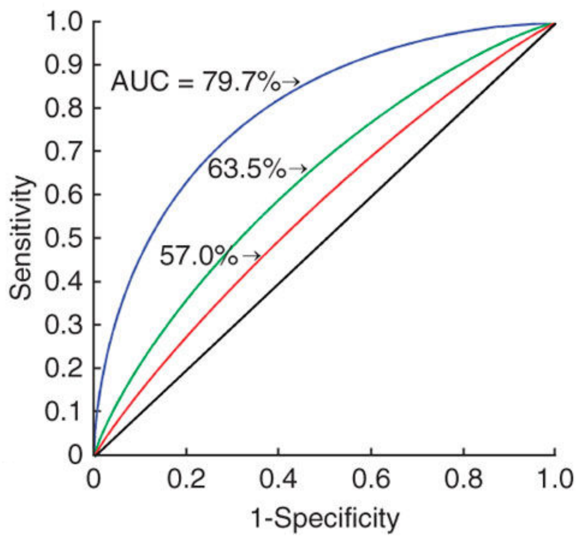


FIGURE 3.3: ROC curves (Nyman-Carlsson et al., 2014)

Figure 3.3 shows different ROC curves with diverse performances (Sing et al., 2005). Sensitivity is identical to TP rate and 1-Specificity is equal to FP rate. Black line refers to a random classifier and the best performance belongs to the blue classifier with the largest AUC. On the other hand, the precision-recall plot (PR) is based on two evaluation measures namely recall and precision as its name suggests. Precision relates to the positive predictions (how many selected instances are relevant) whereas recall refers to the whole positive part of a dataset (how many relevant instances are selected). A neat way to summarize the two aforementioned measures with respect to the confusion matrix metrics follows:

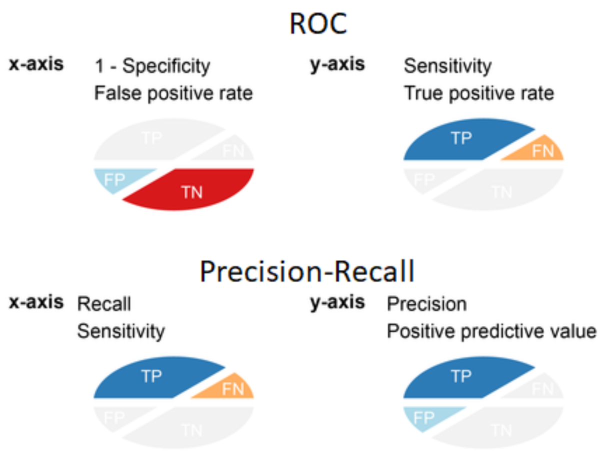


FIGURE 3.4: ROC and PR (Saito and Rehmsmeier, 2016)

There is a one-to-one relationship between ROC and precision-recall points proved by Davis and Goadrich (Davis and Goadrich, 2006). That is, one point in the precision-recall space always has a corresponding point in the ROC space, and vice versa. Hence,precision-recall and ROC curve should indicate the same performance level for a classifier. Nonetheless, they can appear different even in interpretation. The fundamental reason that PR is also partly presented in the results is its different perspective; PR and PR - AUC is affected from class imbalance and provide extra information considering the performance and the influence of the imbalanced data. The key characteristic is that ROC takes into account the negative instances too (specificity), while PR focuses only on the positive ones. It should be noted that the baseline performance metric is ROC - AUC value which will be discussed next.

ROC - AUC value

ROC - AUC is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (with the assumption that "positive" ranks higher than "negative") It is derived from the previous metrics as follows:

$$AUC_{ROC} = \frac{1 + TP_{rate} - FP_{rate}}{2} \tag{3.19}$$

More specifically, the area measures discrimination, that is, the ability of the test to correctly classify links that are going to exist in the future and links that will not and in the present thesis is approximated in the following manner in our dataset.

1. Execution of the link prediction algorithm - scores are assigned to every node pair
2. Split instances to positive/negative class from dataset information
3. AUC value is derived from a trapezoid approximation to the data in step 2

Apart from the approximation in step 3, a variation could be the comparison of a large number of pairs of instances (one from each class). The final value would be:

$$AUC_{ROC} = \frac{\beta + 0.5\gamma}{\alpha} \tag{3.20}$$

Where α is the number of independent comparisons, β is the number of time a positive class node pair has a larger value of a negative pair and γ refers to identical values in the

comparison.
In that way, the increased robustness that ROC - AUC provides, can provide more insightful conclusions since the links that appear in a future time are severely less than the ones that belong to the negative class.
In the majority of previous work ROC - AUC statistic is very commonly used. However, it should be noted that due to the fact that ROC - AUC reduces the ROC Curve to a single number and considers the performance of an individual system, it ignores that the curve is strongly connected to the tradeoffs between the different systems or the plotted performance points.

Scenarios for Class Imbalance

A number of classification algorithms were used on the data however SVM yielded the best results and it will be the only one discussed in the next sections. For the novel task of determining the best rewards, class imbalance needs to be taken into account (Japkowicz and Stephen, 2002). That was achieved with 2 methods:

1. Define a special parameter in the SVM algorithm that adjusts weights inversely proportional to class frequencies in the input data. More specifically, it sets the parameter C from (3.16) of class i to class weight[i] $* C$. In order to balance the 2 classes, the algorithm uses the values of y to automatically adjust weights that are inversely proportional to class frequencies. That is:

$$\text{class weight[i]} = \frac{\text{Total Number of Samples}}{\text{Number of classes} * \text{Number of Samples for class i}} \tag{3.21}$$

2. By using clustering we divide the majority class into K distinct clusters no overlap of observations is allowed among these clusters. Next step is to train each of these clusters with all observations from the minority class and then we average the results for the final prediction.

The diagram below is an example:

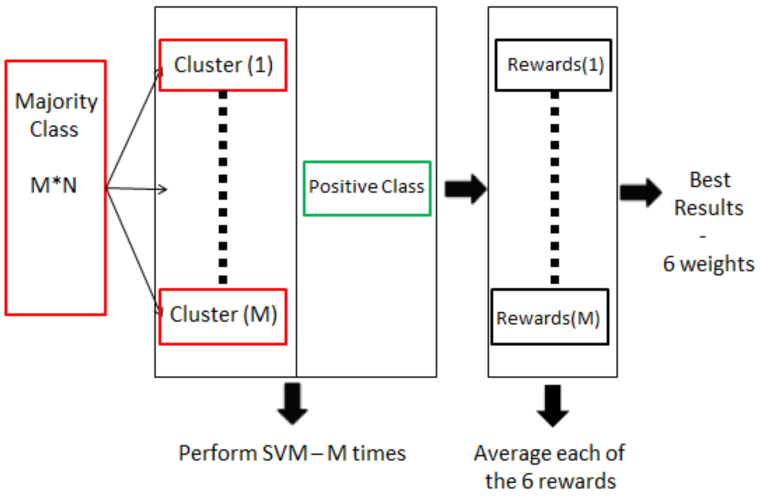


FIGURE 3.5: Clustering technique for Class Imbalance

After a thorough comparison of those 2 methods, it can be concluded that method 1 yielded the best rewards with respect to the suggested performance measure (AUC value).

Chapter 4

Link Prediction in Stack Overflow

Stack Overflow is the first network that is being explored and a piece of initial information is shown below:

- Contains 5 years of interactions
- $\approx$ 1.5 million User ids
- $\approx$ 10 million interactions
- $\approx$ 500000 questions/threads

User ids are the nodes and interactions are the edges of the graph and the sparsity can be proved by the fact that the number of Interactions is close to the number of the nodes and far less than the maximum one.
The network is reduced to contain only 1 year of interactions which will be the largest window of temporal information that can be exploited. Users that have to interact for more than 1 year are not considered relevant as their relationship cannot be characterized as substantial. The analysis that follows refers to the aforementioned subgraph.

4.1 Basic Network Metrics

The subgraph contains 728621 users and 2909010 interactions and some metrics in comparison with a random graph of the same size.

TABLE 4.1: Basic Network Metrics

	Stack Overflow (1 year)	Random Graph (Erdos-Renyi)
Average Degree	7.98	7.4
Global Clustering Coefficient	$2.73 * 10^{-3}$	$1.13 * 10^{-5}$

Table 4.1 illustrates the significant clustering effect of the studied network compared to a random graph. Diameter is also indicative of the small world property that was discussed before. For a sharper description of the network, the following figures are introduced. The scale-free effect can be shown in the figure below.

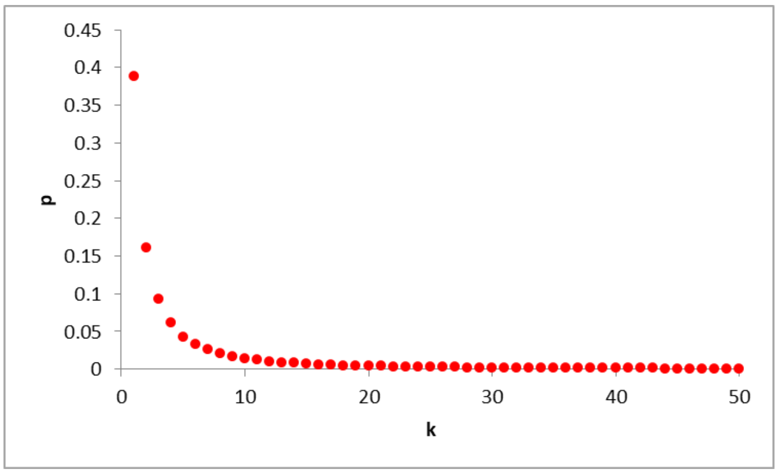


FIGURE 4.1: Degree Distribution of user interactions in Log-Log axes

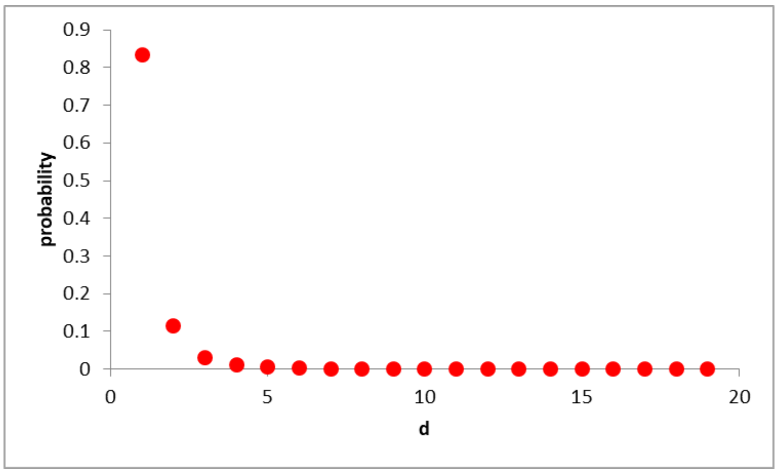


FIGURE 4.2: Histogram of temporal events

The figure 4.1 depicts the power law behavior of the degree distribution of user interactions since the decrease is almost linear. Hence, there are only a few very active users that participate in discussion s while the majority has low activity. Degree distribution is the frequency count of the occurrence of each degree.
Thus, Stack Overflow validates the 3 common characteristics observed in social networks and can be characterized as a typical example of that category. Figure 4.2 reveals the distribution of link appearances during one year. The vast majority of the links, approximately 2/3, appear only once while less than 5% appear more than 3 times in 1 year. As a result, the network cannot be characterized as particularly active and due to that inactivity, the number of temporal events will also be low.

4.1.1 Survival and Waiting time

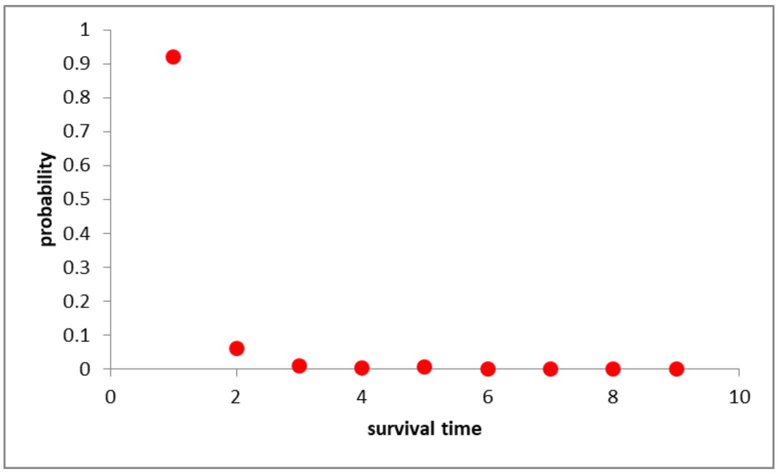


FIGURE 4.3: Survival time of the links

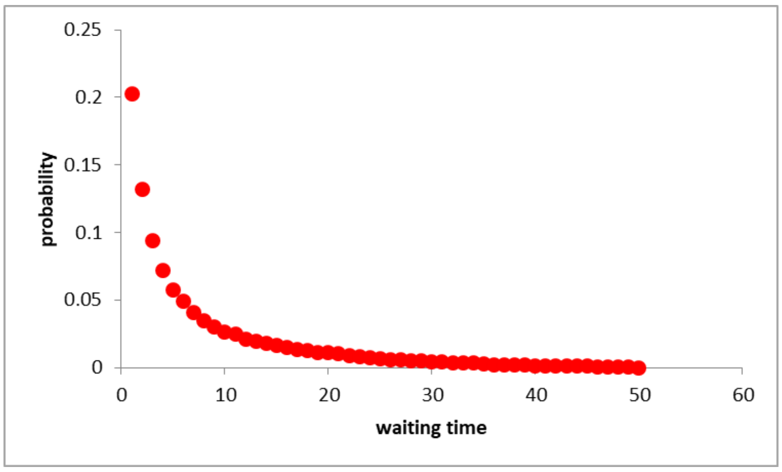


FIGURE 4.4: Waiting time of the links

By using waiting and survival time we would like to investigate the behavior of the links; how long does a link stay active and how long does it take for a link to reappear are two questions to be answered. The survival time of a link is the number of consecutive frames that the link is active. It is important to note that from almost 3 million links only around 30 thousand maintain their appearance for 2 consecutive frames. What is more, the maximum survival time is 9 out of a possible maximum of 51. It would be interesting to see how that weekly temporal activity can assist in possible performance improvement. Waiting time, in that case, refers to the number of consecutive inactive slots between two appearances of a link. More or less, all intermediate possible values are expressed while the extreme case of the right side highlights how rare it is to have two appearances so distant with each other. Left side is more populous, showing that link appearances are more probable to happen close to each other.
It was attempted to fit a distribution in the previous figure and the best results were obtained by an exponential fit and some acknowledged quality metrics are presented below:

TABLE 4.2: Evaluation Metrics for distribution fit

P-value	2.20E-16
Residual Standard Error	0.4222
R-squared	0.9408
F-statistic	763.3

- Small *p-value* indicates that it is unlikely we will observe a relationship between the predictor and response variables due to chance
- The *Residual Standard Error* is the average amount that the response will deviate from the true regression line
- The R-squared statistic provides a measure of how well the model is fitting the actual data. It takes the form of a proportion of variance
- *F-statistic* is a good indicator of whether there is a relationship between our predictor and the response variables. The further the F-statistic is from 1 the better it is.

4.2 Temporal Link Prediction in Stack Overflow

Next step is to perform link prediction deploying the tools discussed in the previous chapter. However, the exact settings for every stage should be clear in order to derive concrete results.

4.2.1 Temporal Structure-Settings

One year of interactions is broken down to 52 consecutive weekly frames. Subsequently, the depth of the temporal information needs to be specified and its behavior needs to be studied. It is crucial to observe how the increase of temporal events can affect the performance of the prediction as well as to highlight the trade-off and find the suitable threshold. Too much information might pose time-efficiency issues not to mention the rational assumption that an old interaction could not bear the same weight as a recent one regarding a future connection.

4.2.2 Link Prediction Scenarios

The following figure illustrates the setup behind the framework.

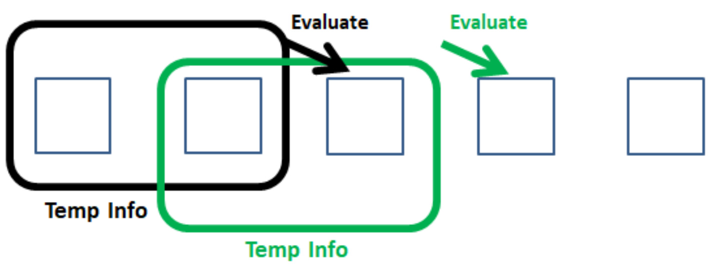


FIGURE 4.5: Setup for 2 weeks of temporal information

The example is reduced in 2 weeks of temporal information but one can assume that all other variations follow the same pattern:

- 2 weeks of interactions form the list of node pairs to be evaluated in a future time

- The suggested proximity measure is deployed to assign scores to the nodes pairs under consideration
- The frame just after the 2 weeks acts as the "future" and is the ground truth of the scenario; the links that appear there are the links that should have the highest proximity score
- Then, as the green box suggests, there is a window shift where the previous steps are repeated. In such a way, the algorithm is evaluated over time by showing how the predefined temporal depth performs over a year of interactions
- Thus, in the specific case, 51 AUC values are derived that summarize the performance of the algorithm

Then, those 5 steps are extended to an arbitrary depth of temporal information that uncovers the network's evolution and quantifies the prediction performance.

4.3 Temporal Link Prediction in Multilayer Networks

In order to further improve the prediction in Stack Overflow the notion of a multilayer network is introduced. In a multilayer network, each type of interaction between the nodes is described by a single layer network and the different layers of networks describe the different modes of interaction. In the present case, common users of Stack Overflow and Github are matched to create the aforementioned network. A brief description of the data used and the design choices are presented below.

4.3.1 Github database

Data were collected with GHTorrent that monitors the Github public event timeline. For each event, it retrieves its contents and their dependencies, exhaustively. It then stores the raw JSON responses to a MongoDB database, while also extracting their structure in a MySQL database. After investigating the different datasets in Github the most meaningful were found to be *pull request history* and *project members* that will be further discussed.

TABLE 4.3: Datasets used in multilayer network

pull request history	project members
id	repo_id
pull_request_id	user_id
created_at	created_at
action	ext_ref_id
actor_id	

The way these datasets were handled was:

- pull request history: Users (*actor_id*) that made a pull request action in the same week (*created_at*) are grouped together and create a temporal network. In other words, if Users "pull" at the same week it is considered to share some sort of similarity.
- project members: When a user (*actor_id*) enters a project (*repo_id*) a link is created that connects him with the rest of the project members at time (*created_at*). Hence, a temporal network is created.

4.3.2 Pull Request

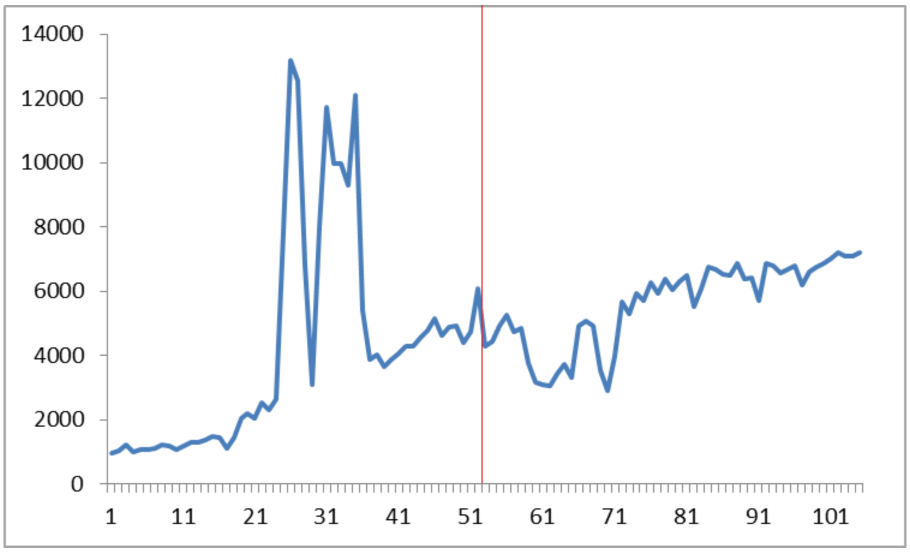


FIGURE 4.6: Pull Request action

min	mean	max
977	4899	13171

The figure above shows the activity of the pull request action in the common users in a period of 2 years (104 weeks). The red vertical line corresponds to the first week of the prediction in Stack Overflow that took place in the previous section. On average, around 5000 actions take place every week and for the year under investigation, it can be seen that the evolution is pretty stable. The number of common users is 17802.

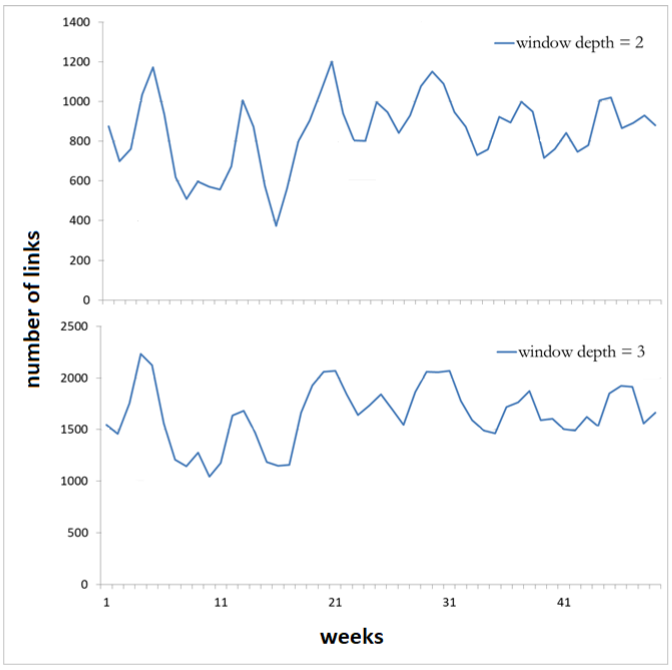


FIGURE 4.7: Links affected for different window depths

TABLE 4.4: Number of links affected from pull request

window depth	min	mean	max
2	373	848.7	1201
3	972	1653	2182

Figure 4.7 depicts how many of the Stack Overflow links are affected by a pull request action every week for two different window depths, as defined previously. The behavior of both subfigures is similar since windows are overlap but it is interesting to observe the doubling in the average links affected by the increase of the window depth.

4.3.3 Project Members

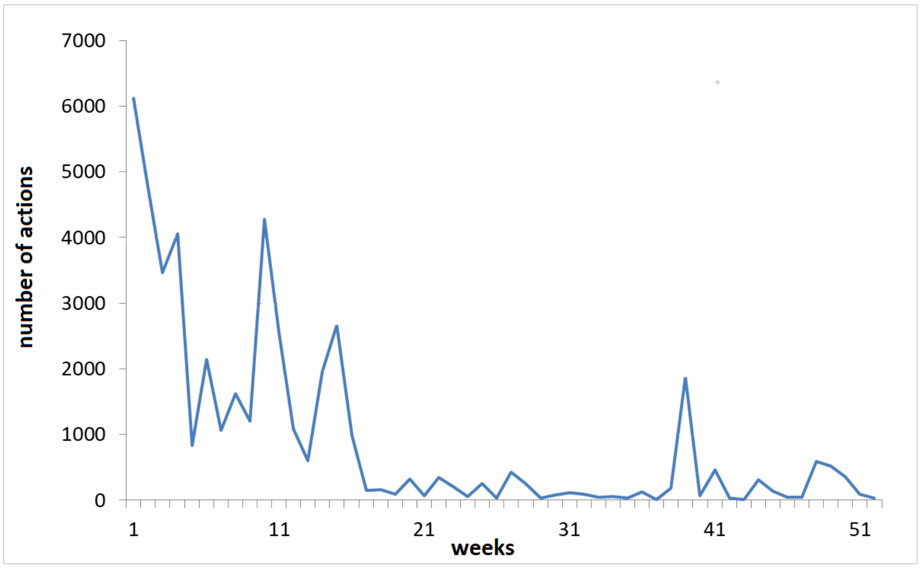


FIGURE 4.8: Activity of project members

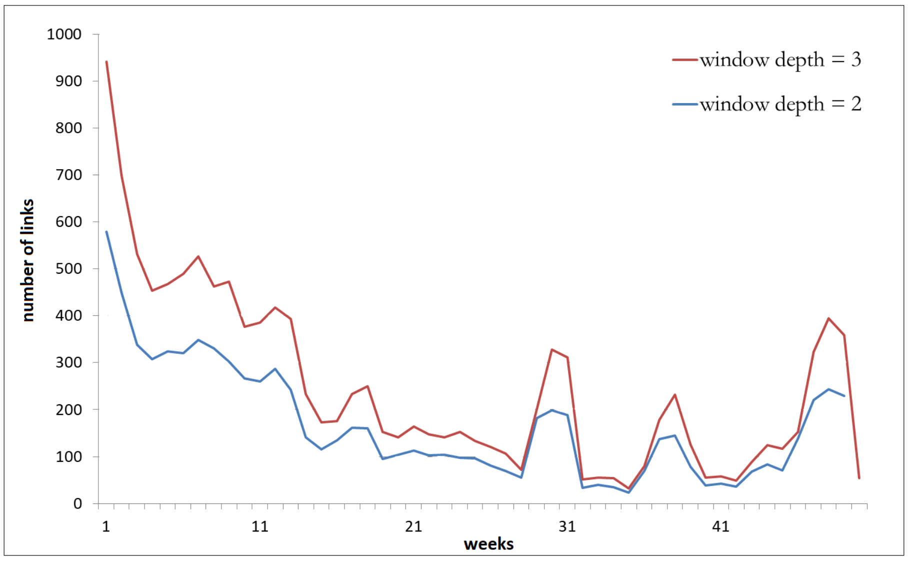


FIGURE 4.9: Links affected for different window depths

Fig 4.8 illustrates the activity of the project members temporal network where users are common between Stack Overflow and Github. The x-axis corresponds to the weekly slices in Stack Overflow and it can be observed that after some weeks the activity becomes very low. The number of common users is 3362 which is significantly lower than the pull request dataset and no significant overlap could be found between links in the two layers. Moreover, 1st and 2nd order neighborhood of the network was investigated to increase the possible link overlap but with no success. Thus, the network was used in a similar fashion as pull request; users that entered a project in the same week are grouped together. Fig 4.9 validates that there is no large difference in the affected links with respect to window depth due to the low activity of the network.

4.3.4 Integration in Machine Learning Algorithm

Those two Github actions are added as features to the SVM algorithm used before to enhance its performance. However, a transformation is required to transform the nodal actions into link characteristics. The following example for pull request action will clarify the above. Same is applied to project members dataset.

- For each node of the node pair under examination count how many pull request actions happened in the designated window depth and combine.
- Combination can be a linear addition, multiplication etc.

To be consistent with the one layered algorithm, scoring as formulated in 3.15 is augmented with the neighborhood actions of the aforementioned Github actions. Wistfully, 1st and 2nd neighborhood did not provide any further activities but since the method is scalable in different design scenarios namely, temporal depth or more social-oriented network the algorithm will succeed to incorporate the neighborhoods.

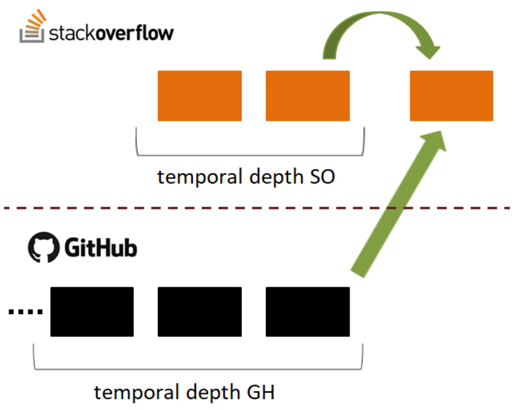


FIGURE 4.10: Representation of Multilayer Link prediction

A thorough investigation for each feature was performed in order to derive the optimal performance. As Fig 4.10 suggests, the temporal depth in Github is not necessarily the same as Stack Overflow. Perhaps a deeper dive in the precedent action yields better performance a notion that is confirmed by the results in the following chapter.

4.3.5 Data split

A fact that is being highlighted in Fig. 4.6 and 4.8 is that very few node pairs possess pull request or project members actions. Github actions (pull request and project members) are present in 1% of the node pairs that are used on the prediction algorithm and an attempt to split the data based on that fact was carried out. Data is split into 2 sets:

- Node pairs with Github actions
- Node pairs with no Github actions

Possibly a test set that consists of active node pairs in the Github layer will yield better results or might provide insight into how the "active" set behaves.

4.4 Statistical Significance Test

Hypothesis testing is a technique for evaluating a theory using data. The “hypothesis” refers to an initial belief and it is known as the alternative hypothesis; the opposite is known as the null hypothesis. More specifically, the prediction results presented in the next chapter are compared by their mean values. Statistical Significance testing indicates whether the difference that is observed in the results corresponds to an actual one in the whole population. If a significant difference between the means is concluded, a safe comparison can take place to identify the superior method accurately.

4.4.1 Methodology

The null hypothesis is that there is no significant difference between specified populations, any observed difference being due to sampling or experimental error. Thus, the alternative hypothesis refers to a substantial difference between the populations. The magnitude of statistical significance is expressed as the p-value. Depending on the statistical test that is chosen, the p-value is the probability of observing the sample results (or more extreme) given that the null hypothesis is true. In other words, this is to consider the probability that a difference in a mean score could have risen based on the assumption that there is actually no difference. E.g, a p-value such as 0.03 means that there is a 3% chance of finding a difference as large as (or larger) given that the null hypothesis is true. Typically, p-values are 5% or less, the null hypothesis is rejected the alternative hypothesis is accepted. The value that is used to reject the null hypothesis is the significance level. Whilst there is relatively little justification why a significance level of 0.05 is used rather than 0.01 or 0.10, for example, it is widely used in academic research. However, since it is desired to be particularly confident in the results, a more stringent level of 0.01 is used. Among the various tests, t-test was used to derive the aforementioned indicators:

$$t-value = \frac{difference\ between\ group\ means}{variability\ of\ groups} = \frac{\bar{X_T}-\bar{X_C}}{SE(\bar{X_T}-\bar{X_C})} \tag{4.1}$$

$$SE(\bar{X_T}-\bar{X_C}) = \sqrt{\frac{var_T}{n_T}+\frac{var_C}{n_C}} \tag{4.2}$$

Next, the computed t-value is looked up on a table of significance to test whether the ratio is large enough to say that the difference between the groups is not likely to have been a chance finding; t-value is mapped to a p-value that is compared to the significance level.The assumptions of a Student’s t-test are:

- Bivariate independent variable (A, B groups)
- Continuous dependent variable

- Each observation of the dependent variable is independent of the other observations of the dependent variable (its probability distribution is not affected by their values). Exception: For the paired t-test, we only require the pair differences to be independent

- Dependent variable has a normal distribution, with the same variance, in each group

However, the t-test used here is Welch's t-test designed for unequal variances and the assumption of normality is maintained. Welch's t-test remains robust for skewed distributions and large sample sizes. Reliability decreases for skewed distributions and smaller samples, where Welch's t-test can only be performed on ranked data. The reason that t-test is deployed and not the also popular z-test is that the standard deviation of the population is unknown.

Chapter 5

Results

5.1 Introduction

In this part, the results with respect to the model presented in Section 3.5 will be given along with the relevant discussion. Section 5.2 will compare the baseline method (Soares and Prudencio, 2013) to the presented ML algorithm and dive into the different temporal depths. Section 5.3 will introduce the second layer (Github) and results of the integrated Multilayer network will be shown. In the end, Section 5.4 provides a summary and suggestions for future steps.

5.2 Link Prediction in Single Layer - Stack Overflow Network

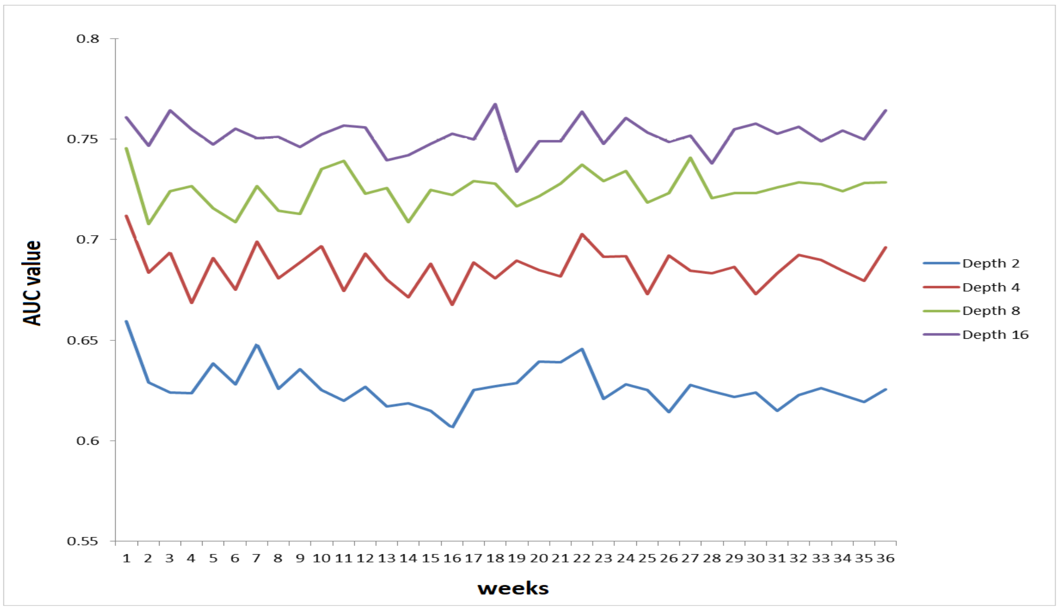


FIGURE 5.1: Performance Curves over time

Fig. 5.1 discusses the improvement of the performance as the temporal depth becomes larger. It shows that increased temporal information leads to better performance and that a significant number of peaks refer to the same "time" for all depths. That signifies a robust enhancement of performance over increasing depth. However, the rate of improvement is not constant; as a matter of fact it decreases when more temporal information is included.

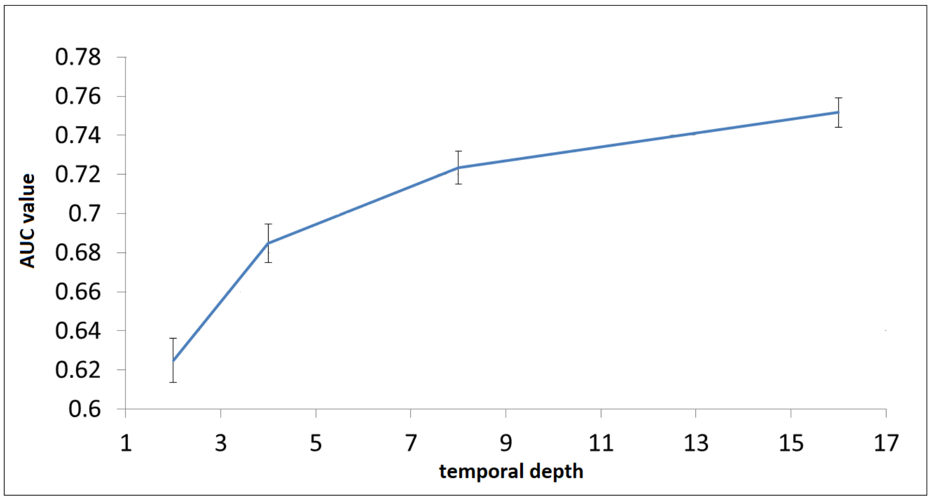


FIGURE 5.2: Performance Curve – Averaged AUC values

In Fig. 5.2, AUC value for each depth is averaged and an error bar is added. That action highlights the decrease of the improvement while depicts that deviation is limited. Thus, the results are stable and the improvement is confirmed.

TABLE 5.1: AUC results for Brute Force and ML

temporal depth	method	mean	max
2	BF	0.6135	0.6489
	SVM	**0.6248**	**0.6594**
	Mean SVM	0.6234	0.6591
4	BF	0.6753	0.7046
	SVM	**0.6848**	**0.7117**
	Mean SVM	0.6846	0.6998
8	BF	0.7153	0.7329
	SVM	**0.7236**	**0.7453**
	Mean SVM	0.7218	0.738
16	BF	0.7446	0.7589
	SVM	**0.7517**	**0.7674**
	Mean SVM	0.749	0.7572

In Table 5.1, for each temporal depth, first row corresponds to brute force that is used in the baseline paper, second is the ML scheme that is introduced here and described previously and third uses fixed rewards that are the mean value of each reward derived from ML over time. Since our method consist essentially of a moving window of a defined temporal length, there are link prediction results for a number of weeks. Thus, max refers to the max AUC value for a specific week and mean refers to the mean value of all the weeks. Evidently, it illustrates the superiority of the ML algorithm compared to the brute force approach used in (Soares and Prudencio, 2013) as in all the cases SVM yields better AUC value that increases with larger temporal history. Another conclusion is that using the mean of the rewards provides very satisfying results. Hence, even if the history of the network is summarized in six fixed results the performance is still acceptable. Lastly, SVM's variance has smaller value although it is low in general. Since the finest results were drawn with SVM, next Section will focus on that and all the upcoming figures refer to the ML method. T-tests were carried out to compare every method and to validate the better performance of the ML algorithm.

TABLE 5.2: T - tests for Brute Force - ML results

temporal depth	2	4	8	16
p - value	6.655e-07	9.907e-06	7.318e-05	0.001

Even with the strictest significance level (0.001) all simulations confirm that SVM produces the finest results.

5.2.1 Performance comparison for different temporal depths

In order to visualize better the behavior of the performance in different temporal lengths the following scatter plots are presented.

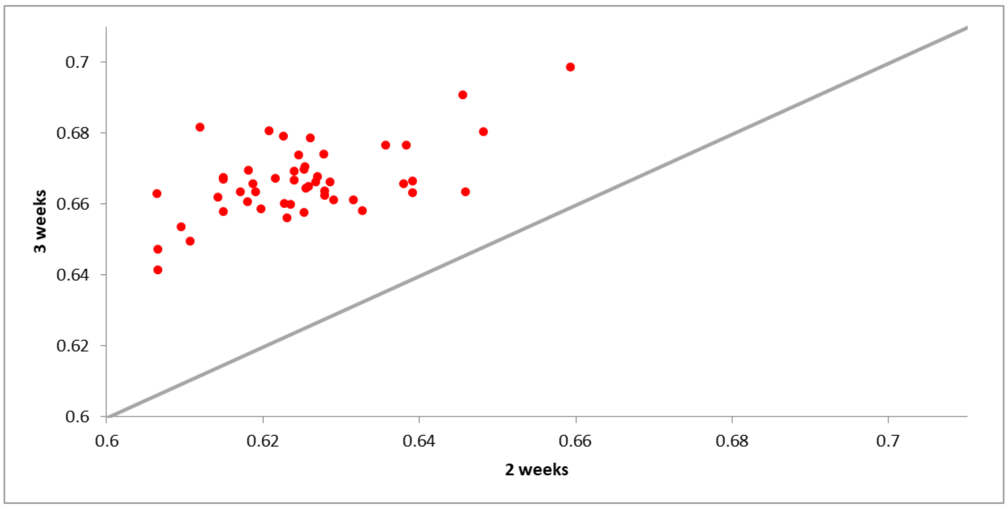


FIGURE 5.3: AUC Scatter plot – depth 2-3

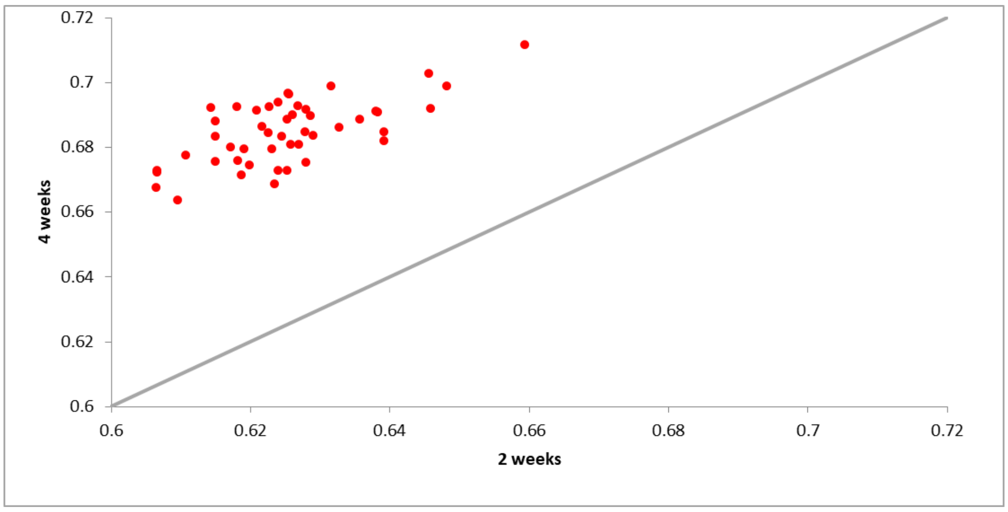


FIGURE 5.4: AUC Scatter plot – depth 2-4

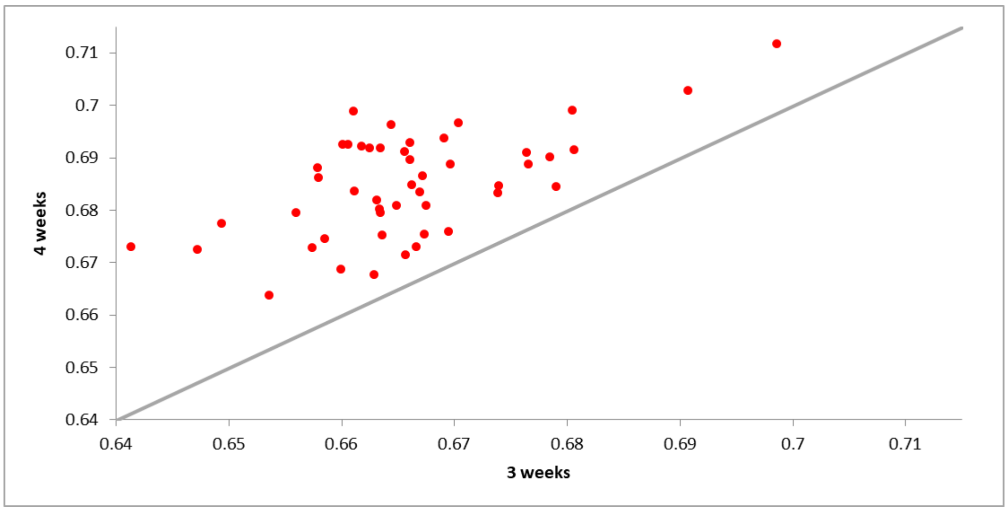


FIGURE 5.5: AUC Scatter plot – depth 3-4

Here, the dots represent AUC values that correspond to the same prediction frame for different depths of temporal information. It is apparent that using 3 weeks of information yields better performance than 2 (Fig. 5.3), and a weak linear trend with relatively heavy variance can be observed; high AUC values on y-axis more or less also provide high relative values on the x-axis. Finally, it can be stated that there is a certain level of stability as high AUC for 2 weeks corresponds to a higher one for 3 weeks at all times. The same behavior can be observed for temporal depths of 2 and 4 with the lone difference of increased performance for four weeks of information. Once more, there is no strong linear correlation but a vague relation can be verified. To conclude, there is definite dominance in the performance by the larger depth of temporal information. Hence, the bigger temporal depth we use the better the performance.

5.2.2 Evolution of the Temporal Events

Machine learning not only provides a better performance regarding AUC value but also allows the observation of the evolution of the rewards over time. That can shed light on what kind of events are considered crucial for the prediction and which of them characterize a specific network.
The rewards are normalized with respect to the innovative reward of the primary events. Thus, the value of the rewards shown below is relative to the creation of a link for the node pair under consideration.

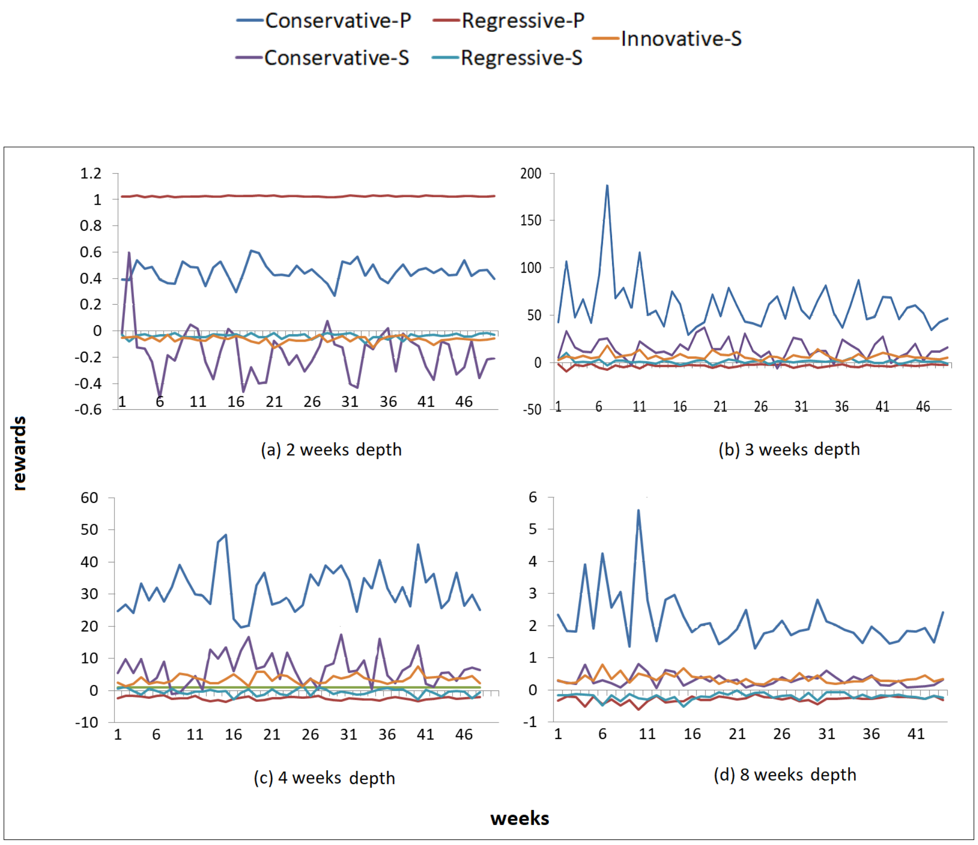
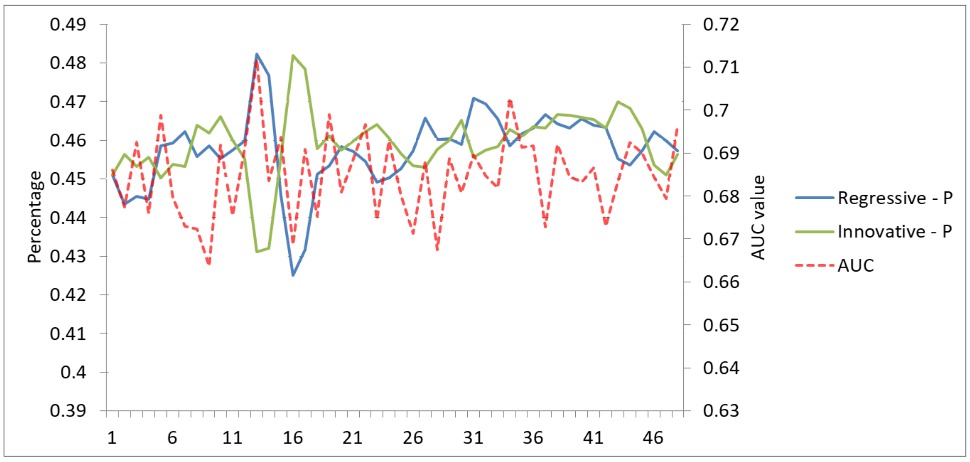


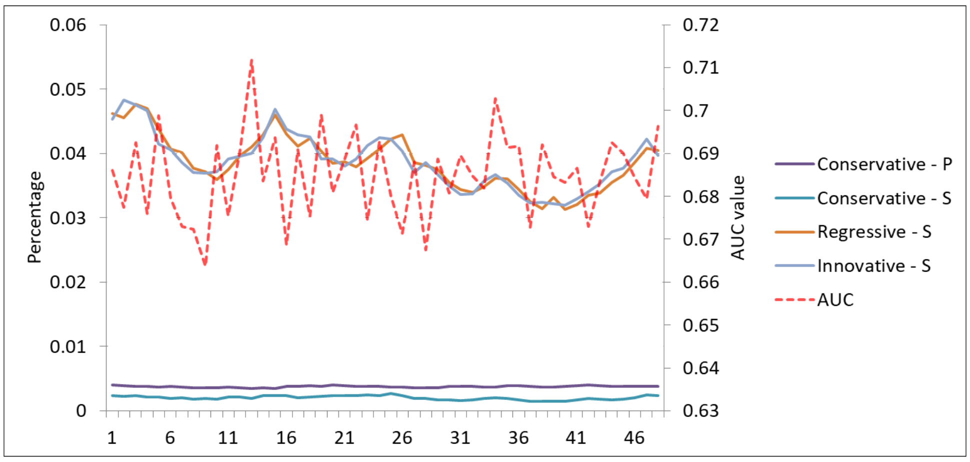
FIGURE 5.6: Evolution of best rewards

The evolution of the rewards for four different cases is shown in Fig. 5.6. Conservative events bear the most significance although the larger the depth the smallest the difference from the rest of the rewards. Only in a) the ML algorithm weights the regressive event as highest and with almost identical value as the innovative event that is not depicted due to the normalization. The variance is notably large, however, it is reduced with depth increase. An elegant validation of the logic behind events is the negative values for both regressive rewards. As they describe the removal of links, thus cancellation of a relationship, they are weighted negatively.

In general, Regressive and Innovative events of the Primary group comprise the majority of the temporal events. The sparsity of the Conservative events could signify their importance; the percentage of them is less than 0.1% for all cases.



(A) 2 weeks



(B) 3 weeks

FIGURE 5.7: Evolution of the percentage of temporal events and the corresponding AUC value - depth 4

Fig. 5.7 visualizes the previous statement regarding the dominance of Regressive and Innovative events. In addition, it was attempted to correlate visually those percentages with AUC values. Unfortunately, no clear correlation can be justified. However, at the subfigure (A), a correlation over time can be observed between Regressive and Innovative events due to the small number of Conservative events, more specifically; links either appear or disappear and hardly remain connected over time. Since visual representations fail to describe any trend that affects the performance, linear and rank correlations of possibly related parameters were derived and shown in a subsequent paragraph.

5.3 Link Prediction in Multilayer Network

5.3.1 Activity Distribution - Github

The way that Github actions are implemented in the algorithm is discussed before. For pull request action the number of common users is 17802 while for project members the number decreases to The number of common users is 3362 and for a finer understanding of each action, it is essential to examine their distribution.

TABLE 5.3: Percentage of node pairs with only one action

temporal depth	2	3
pull request	0.988	0.983
project members	0.998	0.997

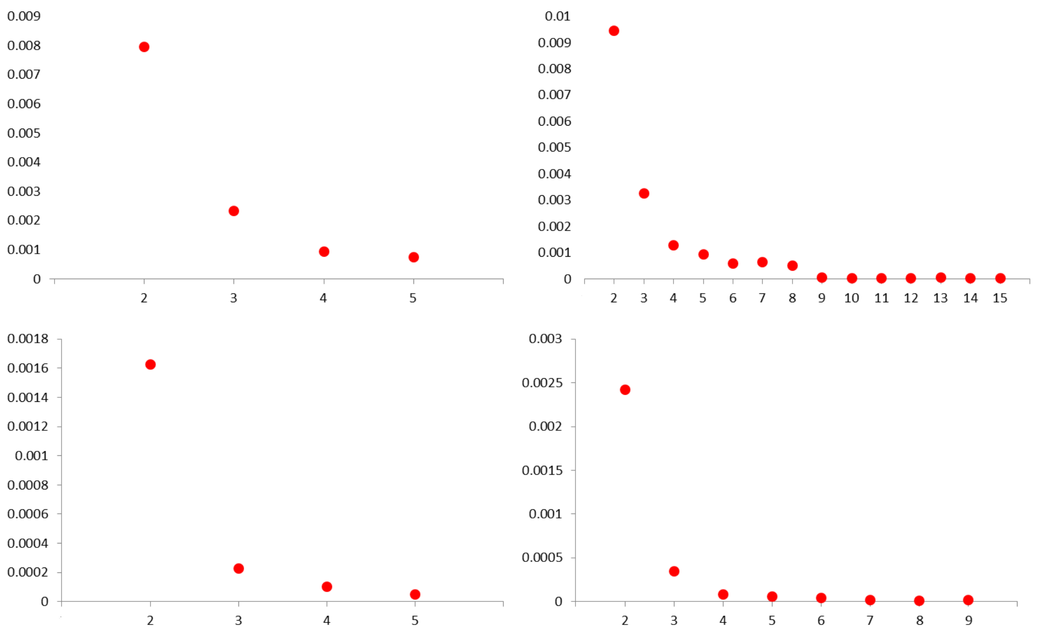


FIGURE 5.8: Activity distribution - Top row (Pull Request), Bottom row(Project Members)

In the Fig. 5.8, x-axis counts how many actions were performed by the node pair and y-axis the percentage of node pairs that have the same number of actions (mean value). The mean value was calculated in the same manner as the performance metric before; every target week provides different feature distribution. Node pairs that had only one action through the studied period consist the majority and are omitted for visualization purposes. By including more temporal information the percentage is slightly falling yet it is apparent that the vast majority of the node pairs perform only one action. It can be observed that pull request is more active and the trend of the decay is more visible.

5.3.2 Performance measurements with the incorporated Github features

The results produced by the integration of the second layer (*Github*) in the target layer (*Stack Overflow*) are exhibited below.

TABLE 5.4: AUC value for temporal depth = 2 in SO

AUC	mean	max	best temp. depth
SO	0.624	0.659	-
lin.comb	**0.63**	**0.668**	**3**
pull request	0.627	0.664	2
project members	0.625	0.661	3
lin.comb-product	0.624	0.66	3

TABLE 5.5: AUC value for temporal depth = 3 in SO

AUC	mean	max	best temp. depth
SO	0.662	0.694	
lin.comb	**0.671**	**0.715**	**5**
pull request	0.666	0.699	5
project members	0.665	0.695	5
lin.comb -product	0.664	0.684	5

TABLE 5.6: T test for Tables 5.4 - 5.5

temporal depth	2	3
p - value	0.011	0.013

In both cases information from Github improved the initial prediction. More specifically, it is shown that the best setting is the linear combination of both actions. Nevertheless, most of the alternatives outperform the single-layer approach. Best temporal depth refers to the fact that in the Github layer, we try to experiment with different temporal depths for each fixed temporal depth in SO. In table 5.4, the best temporal depth ends up to be 3, proving that incorporating more information is beneficial. In 5.5 the depth is increased and equal with 5 for all the scenarios. Thus, it is reasonable to say that depending on the network under consideration, a calibration of the above parameter should take place to take the most of the multilayer scheme. In the end, the performance in both cases is improved by 10%. Again, the consistency of the results is proven through the t-test. Although, they are not lower than the stringiest threshold the improvements are considered legitimate.

Results for split data

The following tables correspond to the two sets; as described in Section 4.3.5. We split the node pairs of SO into two sets; set 1 refers to the node pairs that also include Github actions and Set 2 consists of node pairs that do not possess information from the Github layer.

TABLE 5.7: AUC value for temporal depth SO = 3

AUC	mean	max
pull request - Set 1	0.6279	0.8269
PR & PM - Set 1	0.6492	0.8343
PR & PM - Set 2	0.6258	0.6625

TABLE 5.8: AUC value for temporal depth SO = 3

AUC	mean	max
pull request - Set 1	0.6679	0.8638
PR & PM - Set 1	0.6844	0.8403
PR & PM - Set 2	0.6670	0.6995

TABLE 5.9: T - test for Tables 5.7 - 5.8 - Pull Request & Project Members

temporal depth	2	3
p - value	0.07	0.06

Even though mean values for Set 1 have risen in both cases, there is significant noise and no clear improvement can be stated judging by the p - values. PR & PM implies both pull request actions and project members actions while possibly, an increased temporal depth could be beneficial for tackling the noise in such a case, as more node pairs will be taken into account.

Using Precision - Recall and ROC Curves to Measure Performance

To illustrate the severe imbalance of the dataset a table based on Precision - Recall follows.

TABLE 5.10: AUC value for Precision - Recall Curves

temporal depth	stack overflow	multilayer	random
2	0.018	0.019	0.05
3	0.020	0.021	

Table 5.10 depicts the average performance of the classifiers in a different way, namely average precision. Class imbalance has a straightforward effect on precision and the random classifier can be set as an example. In ROC - AUC value, a random classifier yielded 0.5 no matter the imbalance as True Negatives dominate while PR - AUC focuses on the positive data and prediction therefore random classifier yields 0.005 (stating that imbalance ratio is around 1:200). Aside from the fact that the values of multilayer networks are again higher, it is noteworthy to observe the overall 4-fold improvement compared to a random classifier.

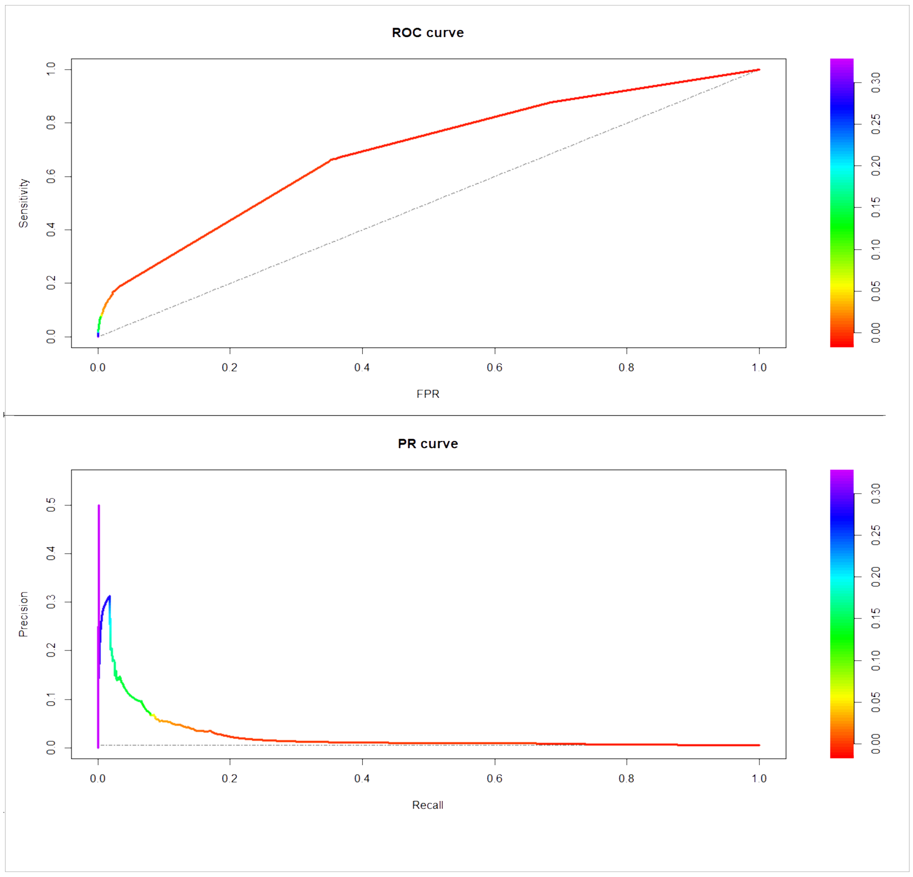


FIGURE 5.9: ROC and Precision-Recall curves for temporal depth - 3

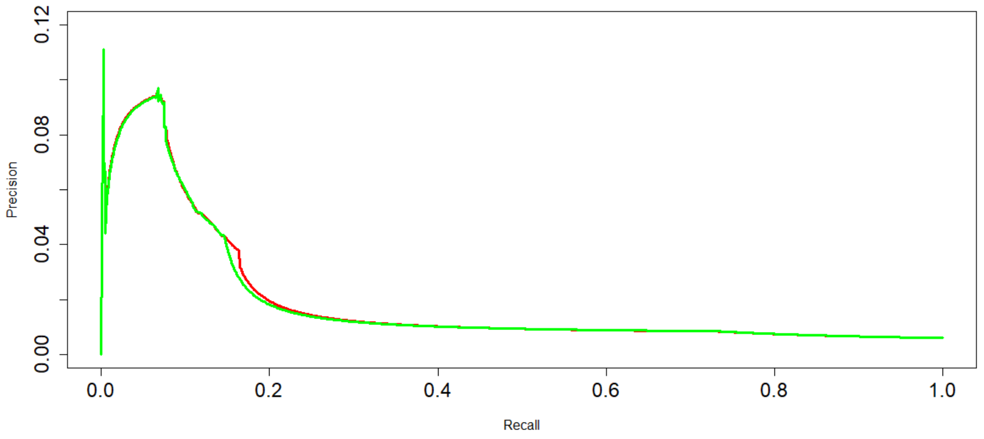


FIGURE 5.10: Precision-Recall - green = single layer, red = multilayer network

Fig. 5.9 displays the curves for all possible thresholds that are used to characterize a positive or negative instance. In our case, a positive instance is a node pair that became a link while a negative instance is a node pair that did not. The color of each point resembles the threshold chosen to achieve that exact behavior and the dotted lines are the random classifiers in each case. Firstly, the scoring range is between 0 - 0.3 and the thresholds are clearer represented on the PR curve as expected. Focusing on the ROC curve, in the early stages (high specificity) there is a promising start but as the threshold is getting lower to include more of the positive instances False Positives are increasing. PR Curve has a noisy start, however, bearing in mind the large imbalance it labels a lot more precise than the random classifier (precision) until it covers the 20% of the positive instances. Fig. 5.10 illustrates the difficulty of retrieving the positive instances accurately and the main concern is the vast number of negative instances are misclassified. Even though the differences are not large, for equal recall (same percentage of selected relevant items) precision is better for multilayer networks. In other words, for a specific number of node pairs belonging to positive class (node pairs that became links), the multilayered approach will label them better.

5.3.3 Correlation between Temporal Events and AUC value

TABLE 5.11: Mean weights of the coefficients of ML algorithm

temporal depth	Cons. - P	Regr. - P	Innov. - P	Cons. - S	Regr. - S	Innov. - S
2	0.45	1.02	1	-0.17	-0.03	-0.06
3	61.1	-3.67	1	14.01	-0.94	6.36
4	31.3	-2.38	1	6.64	-0.45	3.55

Table 5.11 shows the weights that SVM assign to each temporal event for different depths. The algorithm considers the Conservative events the most important for the prediction while the larger the depth the smaller the differences between the weights. Bearing that in mind, it is attempted to show some relation between Conservative events and the performance of the algorithm. and the most straightforward way is to measure their correlation. Here, two types of correlation are investigated Pearson product moment correlation and

Spearman rank-order correlation. The Pearson correlation calculates how strong is the linear relationship between two continuous variables, that is when a change in one variable is associated with a proportional change in the other variable.

$$\text{Pearson correlation} = r = \frac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\overline{x})^2(y_i-\overline{y})^2}} \tag{5.1}$$

where x_i and y_i are the variables and $\overline{x},\overline{y}$ are the corresponding average values. The Spearman correlation calculates the monotonic relationship between two ordinal or continuous variables.The Spearman correlation coefficient is not based on the raw data but on the ranked values for each variable. In our case, we can use Spearman to investigate whether the total amount of events has an impact on the AUC value.

$$\text{Spearman correlation} = \rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \tag{5.2}$$

n is the number of samples and d is the pairwise distances of the ranks of the variables x_i and y_i.

Correlation of total number of temporal events

TABLE 5.12: Correlation between total number of temporal events and AUC

temporal depth	corr. type	Cons - P	Regr - P	Inn - P	Cons - S	Regr - S	Inn - S
2	linear	0.35	0.01	0.2	0.04	0.03	0.11
	rank	0.41	0.1	0.01	0.05	0.06	0.1
3	linear	0.36	0.17	0.05	0.04	0.05	0.02
	rank	0.55	0.24	0.21	0.13	0.05	0.01
4	linear	0.39	0.34	0.31	0.05	0.12	0.11
	rank	0.66	0.38	0.38	0.05	0.12	0.12

TABLE 5.13: Mean weights of Github features

temporal depth	pull request	project members
2	0.6	0.9
3	0.24	1.5
4	0.26	1.9

TABLE 5.14: Correlation between total number of temporal events and AUC - Github

temporal depth	correlation type	pull request	project members
2	linear	0.05	0.2
	rank	0.1	0.25
3	linear	0.07	0.26
	rank	0.14	0.29
4	linear	0.08	0.27
	rank	0.13	0.31

The values represent the correlation coefficients between the total number of the corresponding temporal event and AUC value. From table 5.12 a moderate correlation exists between Conservative events and AUC value which grows larger with the temporal depth. So it can be cautiously said that the more Conservative events there are, the better the prediction justifying the bigger weight that ML assigned to the Conservative event. Furthermore, in larger temporal depths, other temporal events become more relevant but always Conservative has a superior correlation. While the results above belong to the 1-layer, it is interesting to observe how the Github features are weighted. Hence, they are comparable with the 1-layer ones although they cannot be considered extra significant. The level of their significance is also validated by the correlation coefficients. Project members seem more important probably due to their rarity; there is fewer project members activity than pull request as shown before. Once more, for the calculation of the correlation coefficients, the total number of each activity is used.

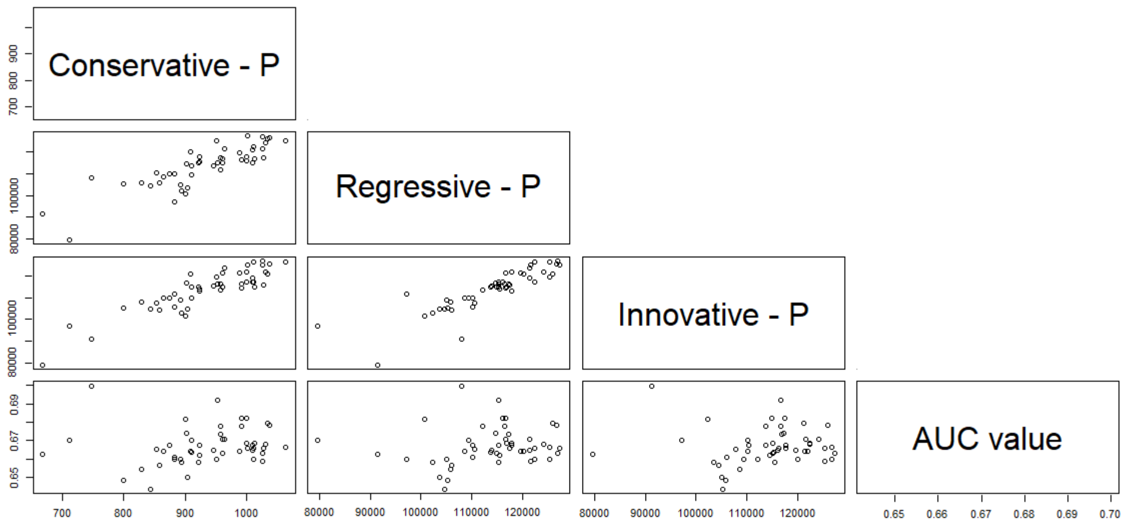


FIGURE 5.11: Scatter plots of Primary temporal events and AUC value

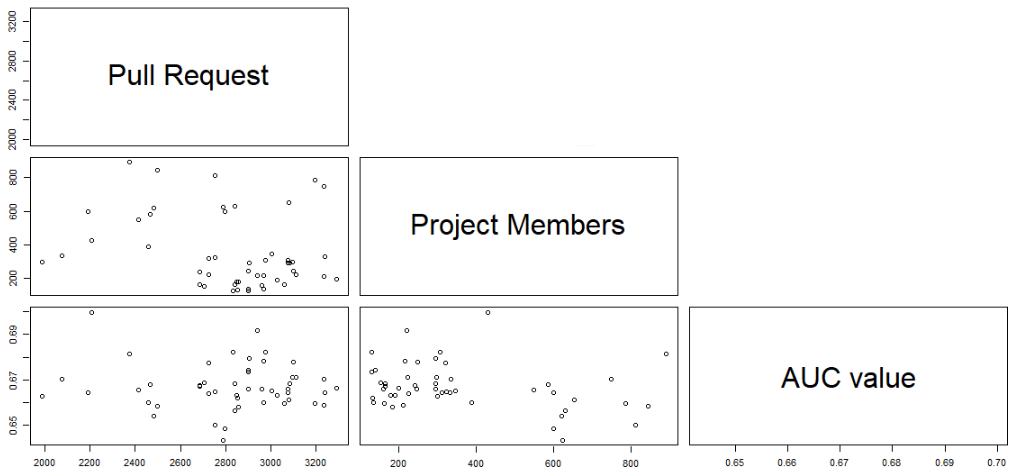


FIGURE 5.12: Scatter plots of Github events and AUC value

Fig. 5.11 focuses on the Primary events and attempts to graphically show potential correlations that cannot be captured by the linear or rank coefficient. The linear trend among them can be easily explained since the total number of events is plotted; In more active weeks all of the events will be more on average. On the other hand, when events are compared with AUC value they do not seem to follow any obvious non-linear trends. Fig. 5.12 focuses on the Github layer and unlike the Primary events, Github features are not correlated by total number. As expected the scatter plots with AUC can be characterized as almost random.

Correlation of the percentage of temporal events

In this subsection, the correlation is calculated between the AUC value and the percentages of each event at the corresponding time. The main thought was to investigate whether the composition of the events each week is correlated to the performance. E.g. In a week that Conservative events are 5% of the total events, is the performance better than a week that the value is 2% ?

TABLE 5.15: Correlations between AUC and percentage of temporal events

temporal depth	corr. type	Cons - P	Regr - P	Inn - P	Cons - S	Regr - S	Inn - S
2	linear	0.22	0.19	0.2	0.009	0.02	0.05
	rank	0.3	0.1	0.04	0.045	0.07	0.1
3	linear	0.27	0.22	0.21	0.17	0.21	0.14
	rank	0.34	0.19	0.26	0.087	0.29	0.14
4	linear	0.37	0.31	0.24	0.21	0.29	0.28
	rank	0.4	0.33	0.31	0.19	0.30	0.25

The table 5.15 indicates some similarity with the correlations of the total number of events. Conservative events appear more correlated and once again a larger depth increases the correlation of the other events.

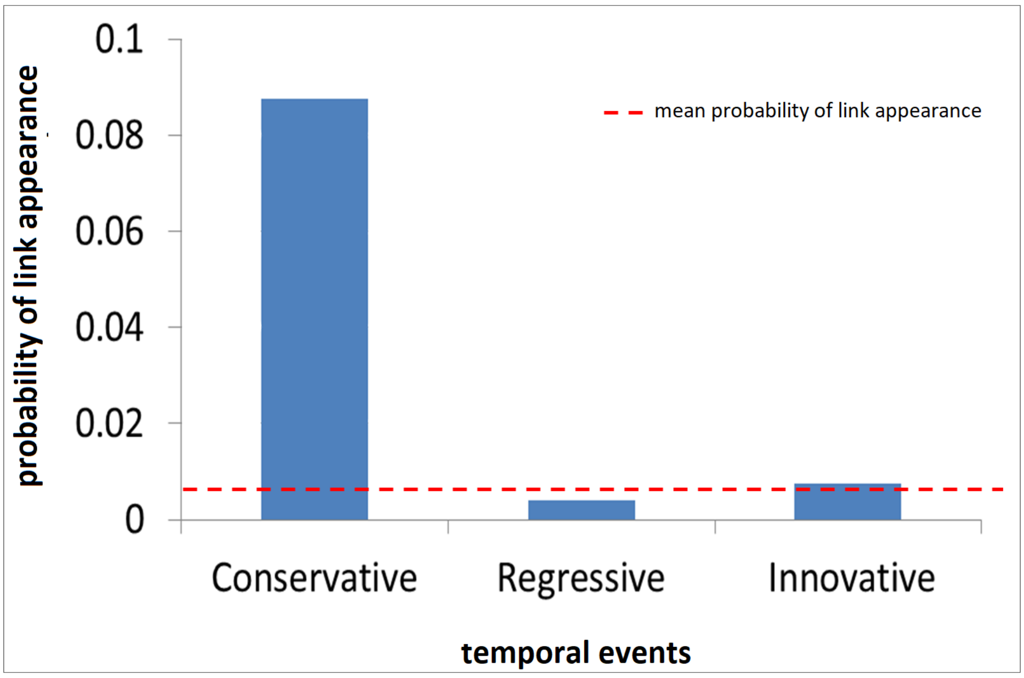


FIGURE 5.13: Probability of Link Appearance for each Primary temporal event

Fig. 5.13 validates the importance of Conservative events. By taking into account the activity of all node pairs, if there is a Conservative event there is around 9% probability that there will be a link next week. Compared to the other Primary events its superiority is apparent. What is more, the red dotted line represents the mean probability of a link appearance if there is some activity (some event) in the weeks under investigation.

5.4 Limitations & Contributions

This last section sums up the contributions of the current thesis on the topic of temporal link prediction and addresses the limitations of the method used in this project.

5.4.1 Limitations

A fundamental limitation was the datasets that were used. Perhaps with networks that are more active and dynamic like Twitter we could exploit in a larger extend the temporal characteristics. Additionally, the overlap between layers was small, thus the multilayer approach can and should be further investigated with data from different sources. In some cases the variation of the results was large and we could not derive concrete conclusions. In general, the model accepts arbitrary networks and the realization of a multilayer network depends strongly on the selected ones. It is always challenging to select the appropriate metrics to measure the performance in imbalanced data. Furthermore, the link prediction was based on a linear combination of temporal events, however, the problem could be formulated using non-linear ones. Linear combination was a choice based on simplicity and due to the fact that it was a neat input for the ML algorithms that were used.

5.4.2 Contributions

While there is a lot of previous work done on link prediction, the temporal aspect and dynamics of the network under investigation are usually ignored. In addition, entities are linked with diversified types of relations that are important factors of the dynamics of the network. The integration of temporal information and different types of interactions, namely the creation of a temporal multilayer network, is a novel tool for link prediction. As the results illustrate, the method is beneficial and superior from the static and single layer approaches. The thesis contributions can be summarized as follows:

- The creation of a temporal multilayer network improves link prediction as it can be seen in Table 5.1 for one layer comparison and Table 5.7 and Table 5.8 for 2 layers.
- The larger the temporal window the better the performance. Fig. 5.2 shows the decreasing improvement for larger temporal windows.
- The most important is the Conservative event. The preservation of a link is the most crucial feature could signify a future link. The result of the algorithm matches the behavior in Fig. 5.13 and supports the statement.
- The deployed ML algorithm is faster than brute force and provides the required weights for separating the two classes. Brute force that is commonly used could fail to find the global maximum for the determination of the weights.
- Using multilayer network in the prediction by linearly combining Github features never provides worse results. In some cases, from Table 5.7 - 5.8 is at least identical to single layer.

5.4.3 Future Work

- Essentially, by introducing the evolution of temporal events the framework can be used to also characterize the user's behavior over time. Regardless of the networks used and apart from link prediction task, the method sheds light on the type of interactions between users through a weighting scheme. By observing the evolution of the temporal events one could derive useful information from the values or a peak/anomaly in the typical time flow. Hence, the framework can be used as an anomaly detector in the future.
- On a higher level, the increasing amount of data that users generate and the diverse networks that they participate will provide information to improve our predictions and possibly highlight the underlying relationship between the different layers.
- The current work can also be used as recommendation system that can be generalized in the future. For example, which people to follow, posts to read or products that might be relevant to buy.
- Machine Learning offers many opportunities to improve the prediction. For example, along with the temporal events, common static methods (e.g. Common Neighbors, Jaccard Index) can be used as features of the algorithm.
- The method is very much aligned to the real world social phenomena. People become more and more part of diverse social networks not to mention the inherently multiple nature of user interactions. Therefore, it is intriguing to research on community detection and the interdependence of the various layers. In that way we can reduce the information needed to perform efficient and accurate link prediction.
- Aside from the social network, the method can be used in different domains like bioinformatics and e-commerce. Relevant example could be a multilayer network of gene interactions where each layer could represent different tissue.

Bibliography

Adomavicius, Gediminas and Alexander Tuzhilin (2005). “Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions”. In: *IEEE transactions on knowledge and data engineering* 17.6, pp. 734–749.

Al Hasan, Mohammad and Mohammed J Zaki (2011). “A survey of link prediction in social networks”. In: *Social network data analytics*. Springer, pp. 243–275.

Al Hasan, Mohammad et al. (2006). “Link prediction using supervised learning”. In: *SDM06: workshop on link analysis, counter-terrorism and security*.

Albert, Réka and Albert-László Barabási (2002). “Statistical mechanics of complex networks”. In: *Rev. Mod. Phys.* 74 (1), pp. 47–97. DOI: 10.1103/RevModPhys.74.47. URL: https://link.aps.org/doi/10.1103/RevModPhys.74.47.

Albert, Réka and Albert-László Barabási (2002). “Statistical mechanics of complex networks”. In: *Reviews of modern physics* 74.1, p. 47.

Asaduzzaman, Muhammad et al. (2013). “Answering questions about unanswered questions of stack overflow”. In: *Proceedings of the 10th Working Conference on Mining Software Repositories*. IEEE Press, pp. 97–100.

Backstrom, Lars et al. (2012). “Four degrees of separation”. In: *Proceedings of the 4th Annual ACM Web Science Conference*. ACM, pp. 33–42.

Barabási, Albert-László and Réka Albert (1999). “Emergence of scaling in random networks”. In: *science* 286.5439, pp. 509–512.

Chen, Zhangxin, Guanren Huan, and Yuanle Ma (2006). *Computational methods for multiphase flows in porous media*. Vol. 2. Siam.

Davis, Jesse and Mark Goadrich (2006). “The relationship between Precision-Recall and ROC curves”. In: *Proceedings of the 23rd international conference on Machine learning*. ACM, pp. 233–240.

De Domenico, Manlio et al. (2013). “Mathematical formulation of multilayer networks”. In: *Physical Review X* 3.4, p. 041022.

Dhote, Yugchhaya, Nishchol Mishra, and Sanjeev Sharma (2013). “Survey and analysis of temporal link prediction in online social networks”. In: *Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on*. IEEE, pp. 1178–1183.

Fawcett, Tom (2004). “ROC graphs: Notes and practical considerations for researchers”. In: *Machine learning* 31.1, pp. 1–38.

Girvan, Michelle and Mark EJ Newman (2002). “Community structure in social and biological networks”. In: *Proceedings of the national academy of sciences* 99.12, pp. 7821–7826.

Hackeling, Gavin (2014). *Mastering Machine Learning with scikit-learn*. Packt Publishing Ltd.

Holme, Petter and Jari Saramäki (2012). “Temporal networks”. In: *Physics reports* 519.3, pp. 97–125.

Homans, George C (1958). “Social behavior as exchange”. In: *American journal of sociology* 63.6, pp. 597–606.

Japkowicz, Nathalie and Shaju Stephen (2002). “The class imbalance problem: A systematic study”. In: *Intelligent data analysis* 6.5, pp. 429–449.

Katz, Leo (1953). “A new status index derived from sociometric analysis”. In: *Psychometrika* 18.1, pp. 39–43.

Kivela, Mikko et al. (2014). “Multilayer networks”. In: *Journal of Complex Networks* 2.3, pp. 203–271.

Kurant, Maciej and Patrick Thiran (2006). “Layered complex networks”. In: *Physical review letters* 96.13, p. 138701.

Liben-Nowell, David and Jon Kleinberg (2007). “The link-prediction problem for social networks”. In: *journal of the Association for Information Science and Technology* 58.7, pp. 1019–1031.

Lü, Linyuan and Tao Zhou (2011). “Link prediction in complex networks: A survey”. In: *Physica A: statistical mechanics and its applications* 390.6, pp. 1150–1170.

Lü, Linyuan et al. (2015). “Toward link predictability of complex networks”. In: *Proceedings of the National Academy of Sciences* 112.8, pp. 2325–2330.

Lytras, Miltiadis D et al. (2010). *Knowledge Management, Information Systems, E-Learning, and Sustainability Research: Third World Summit on the Knowledge Society, WSKS 2010, Corfu, Greece, September 22-24, 2010, Proceedings*. Vol. 111. Springer.

Millen, David R and Jonathan Feinberg (2006). “Using social tagging to improve social navigation”. In: *Workshop on the Social Navigation and Community based Adaptation Technologies*. Citeseer.

Murata, Tsuyoshi and Sakiko Moriyasu (2008). “Link prediction based on structural properties of online social networks”. In: *New Generation Computing* 26.3, pp. 245–257.

Newman, Mark EJ (2001). “Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality”. In: *Physical review E* 64.1, p. 016132.

Nyman-Carlsson, Erika et al. (2014). “Eating Disorder Inventory-3: Validation in Swedish patients with eating disorders, psychiatric outpatients and a normal control sample”. In: *Nordic Journal of Psychiatry* 68, pp. 1–10. DOI: 10.3109/08039488.2014.949305.

Oyama, Satoshi, Kohei Hayashi, and Hisashi Kashima (2011). “Cross-temporal link prediction”. In: *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, pp. 1188–1193.

Pedregosa, Fabian et al. (2011). “Scikit-learn: Machine learning in Python”. In: *Journal of machine learning research* 12.Oct, pp. 2825–2830.

Saito, Takaya and Marc Rehmsmeier (2016). *Classifier evaluation with imbalanced datasets*.

Salton, G and MJ McGill (1986). “Introduction to modern information retrieval Google Scholar”. In:

Sayad, Saed (2017). *An Introduction to Data Science*.

Sharma, Shikhar and Anurag Singh (2015). “An efficient method for link prediction in complex multiplex networks”. In: *Signal-Image Technology & Internet-Based Systems (SITIS), 2015 11th International Conference on*. IEEE, pp. 453–459.

Sing, Tobias et al. (2005). “ROCR: visualizing classifier performance in R”. In: *Bioinformatics* 21.20, pp. 3940–3941.

Soares, Paulo RS and Ricardo BC Prudencio (2013). “Proximity measures for link prediction based on temporal events”. In: *Expert Systems with Applications* 40.16, pp. 6652–6660.

Strogatz, Steven H (2001). “Exploring complex networks”. In: *nature* 410.6825, p. 268.

Tabourier, Lionel, Anne-Sophie Libert, and Renaud Lambiotte (2016). “Predicting links in ego-networks using temporal information”. In: *EPJ Data Science* 5.1, p. 1.

Travers, Jeffrey and Stanley Milgram (1967). “The small world problem”. In: *Phychology Today* 1.1, pp. 61–67.

Wasserman, Stanley and Katherine Faust (1994). *Social network analysis: Methods and applications*. Vol. 8. Cambridge university press.

Watts, Duncan J and Steven H Strogatz (1998). “Collective dynamics of ‘small-world’networks”. In: *nature* 393.6684, p. 440.

Xiang, Evan Wei (2008). “A survey on link prediction models for social network data”. In: *Science and Technology*.

Yang, Jinxuan and Xiao-Dong Zhang (2016). “Predicting missing links in complex networks based on common neighbors and distance”. In: *Scientific reports* 6, p. 38208.