CHARACTERIZATION AND IDENTIFICATION OF SYNONYMS ON ANONYMOUS SOCIAL NETWORKS

Janaína Sant'Anna Gomide Gomes

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientador: Daniel Ratton Figueiredo

Rio de Janeiro
Maio de 2019

CHARACTERIZATION AND IDENTIFICATION OF SYNONYMS ON ANONYMOUS SOCIAL NETWORKS

Janaína Sant'Anna Gomide Gomes

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof. Daniel Ratton Figueiredo, Ph.D.

Prof. Gerson Zaverucha, Ph.D.

Prof. Valmir Carneiro Barbosa, Ph.D.

Prof. Alberto Henrique Frade Laender, Ph.D.

Prof. Bruno Felisberto Martins Ribeiro, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
MAIO DE 2019

Gomide Gomes, Janaína Sant'Anna

Characterization and Identification of Synonyms on Anonymous Social Networks/Janaína Sant'Anna Gomide Gomes. – Rio de Janeiro: UFRJ/COPPE, 2019.

XIII, 84 p.: il.; 29, 7cm.

Orientador: Daniel Ratton Figueiredo

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2019.

Referências Bibliográficas: p. 79 – 84.

1. ambiguity. 2. social networks. 3. algorithm. I. Figueiredo, Daniel Ratton. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Dedico esta tese aos meus pais
que sempre me incentivaram a
estudar e a buscar novos
conhecimentos.

Agradecimentos

Agradeço a Deus por me guiar, iluminar e me ajudar a superar as dificuldades e desafios encontrados ao longo desses anos.

Agradeço ao meu pai, ex-aluno do PESC, que foi um exemplo de pessoa e profissional. Fico feliz de termos cursado o mesmo programa de pós-graduação. Agradeço a minha mãe que está sempre ao meu lado, me ajudando em tudo e sempre me apoiando. Obrigada por me estimular e sempre acreditar em mim. Agradeço a minha irmã pelo carinho e por me incentivar a terminar o doutorado.

Agradeço ao meu marido, meu companheiro de todas as horas, que me apoia e incentiva a buscar o melhor para minha vida profissional. Agradeço à minha filha, que ainda nem nasceu, mas que já me motiva a ser uma pessoa melhor.

Agradeço ao meu orientador, Prof. Daniel, que me orientou de perto durante todo o doutorado e que é meu grande exemplo de professor e orientador. Obrigada pelas reuniões, pelas orientações, pelas revisões e por ter sempre me incentivado a buscar mais conhecimento. Diversas vezes não sabia como continuar ou não estava motivada e depois de reunirmos sempre voltada animada a continuar trabalhando.

Agradeço aos meus amigos e familiares que sempre acreditaram em mim.

Agradeço aos professores e aos funcionários do PESC pelos ensinamentos e suporte ao longo desses anos.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

CARACTERIZAÇÃO E IDENTIFICAÇÃO DE SINÔNIMOS EM REDES SOCIAIS ANÔNIMAS

Janaína Sant'Anna Gomide Gomes

Maio/2019

Orientador: Daniel Ratton Figueiredo

Programa: Engenharia de Sistemas e Computação

Em muitos cenários objetos são referenciados por meio de vários nomes e essa diversidade de nomes gera ambiguidade. Abordar o problema de ambigüidade de nome é um passo importante na consolidação de dados e com o crescimento da quantidade de dados digitais, tornou-se indispensável. Além disso, o contínuo aumento da preocupação com privacidade por parte de indivíduos e empresas está alterando a forma como os dados ficam disponíveis. Em particular, a remoção de informações pessoalmente identificáveis (PII) está se tornando uma prática comum. Nesse trabalho é feita a caracterização e identificação de sinônimos em redes sociais anônimas e somente a estrutura da rede é considerada, toda PII foi removida. As principais contribuições desta tese são classificar os padrões de uso de diferentes nomes pelos indivíduos que possuem múltiplos nomes, propor um modelo probabilístico para sinônimos em redes sociais, e propor algoritmos para identificar sinônimos em redes sociais anônimas. O primeiro algoritmo considera disância entre nós e número de vizinhos em comum para identificar sinônimos em uma rede social. O segundo algoritmo considera perfis de indivíduos em redes de colaboração e identifica diferentes nós que correspondem ao dono do perfil. O algoritmo é baseado no problema do conjunto dominante e conjunto independente em grafos. O último algoritmo é um framework que classifica nós como tendo duplicatas em redes sociais. Esse algorithmo extrai subgrafos para gerar as características que são utilizadas como entrada para rede neural de dois níveis, projetada especificamente para esse problema. Bases de dados reais de redes de colaboração, extraídas do DBLP e Google Scholar, assim como redes de famílias são utilizadas para avaliar os algoritmos propostos. Resultados experimentais indicam que sinônimos podem ser efetivamente identificados mesmo em redes sociais anônimas considerando apenas a estrutura da rede.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

CHARACTERIZATION AND IDENTIFICATION OF SYNONYMS ON ANONYMOUS SOCIAL NETWORKS

Janaína Sant'Anna Gomide Gomes

May/2019

Advisor: Daniel Ratton Figueiredo

Department: Systems Engineering and Computer Science

In many scenarios objects are referred to using multiple labels and this diversity leads to ambiguities. Addressing name ambiguity is an important step in data consolidation and with the growth in the amount of digital data has become even more pressing. Moreover, the growing privacy concerns among individuals and enterprises is leading to the removal of personally identifiable information (PII) in data that is publicly available. In this work, we focus on the characterization and identification of synonyms in anonymous social networks where only the network structure is considered, all PII has been discarded. The main contributions of this thesis are to classify name usage patterns by individuals that use multiple names, to propose a probabilistic model for synonyms in social networks, and to propose algorithms to identify synonyms in anonymous social networks. The first algorithm considers distance between nodes and number of common neighbors to identify synonyms in a social network. The second algorithm considers ego-centered collaboration networks and identifies the different nodes that correspond to the egonet owner. The algorithm is based on the dominating set and independent set problems in graphs. The last algorithm is a framework that classifies nodes as having duplicates in social networks. This algorithm extracts subgraphs to generate features for nodes that are then used as input to a two-level neural network designed specifically for this problem. Real collaboration networks, extracted from DBLP and Google Scholar, as well as familial networks are used to evaluate the proposed algorithms. Experimental results indicate that synonyms can effectively be identified even on anonymous social networks leveraging only network structure.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In many scenarios unique objects are referred to using multiple labels. For example, in the Web a city like New York can be referred to by labels such as, "New York", "NYC" and "Big Apple"; in bibliographical data a person can be referred to with various names, such as "von Neumann", "John von Neumann" and "J. von Neumann". The diversity of labels used to refer to objects leads to ambiguities and is a recurring problem in structured and unstructured data such as the Web and bibliographic datasets. Addressing name ambiguity is an important step in data consolidation and has long been subject of study in academia and industry [2, 3]. With the incredible growth in the amount of digital data in the past years, addressing name ambiguity has become even more pressing.

Collaboration networks encode the structure of the collaborative effort to jointly produce an outcome such a book, a movie or a scientific paper. In such networks, individuals are represented as nodes and pair-wise collaborations are represented by edges. Most collaboration networks are constructed using the names associated with individuals as they appear in a given collaboration recorded in a dataset. Consequently, since such names are usually ambiguous, the constructed network will also have ambiguities. For example, Table 1.1 shows three papers co-authored by John von Neumann in which his name appeared with different spellings such as "J. v. Neumann", "J. von Neumann" and "John von Neumann" (as recorded in the bibliographic records of the journals). The collaboration network induced by these three publications is illustrated in Figure 1.1 where three nodes for John von Neumann are created, each corresponding to a different label (name) associated with the person.

As illustrated above, one of the disambiguation problems in collaboration networks consists of identifying nodes (names) that correspond to the same individual, known as the synonym problem[1], a task far from trivial despite continued efforts in

[1] Another disambiguation problem occurs when a single name is used by different individuals, thus a single network node can represents multiple individuals (homonym problem).

Table 1.1: Three publications of John von Neumann cataloged with different names.

Authors	Title	Journal	Year
B. O. Koopman, J. v. Neumann	Dynamical systems of continuous spectra	Proceedings of the National Academy of Sciences	1932
J. von Neumann, R. H. Kent, H. R. Bellinson, B. I. Hart	The mean square successive difference	The Annuals of Mathematical Statistics	1941
B. I. Hart, John von Neumann	Tabulation of the probabilities for the ratio of the mean square successive difference	The Annuals of Mathematical Statistics	1942

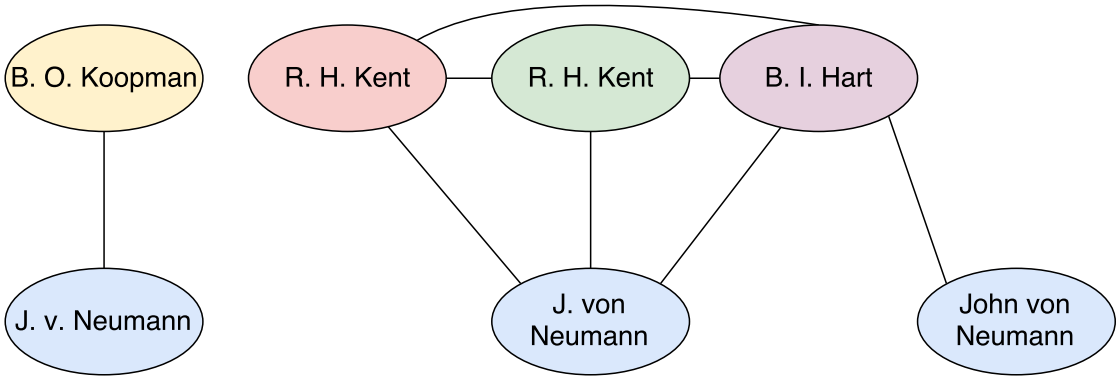


Figure 1.1: Collaboration network induced by three publications of John Von Neumann in Table 1.1 showing three nodes for the same individual.

the literature (see Chapter 2).

Another kind of social network where ambiguities are present is familial networks. In these networks, multiple partial representations of a family tree are provided from the perspective of different family members. For example, different family members when describe their relatives using different names and when the family tree is constructed there are duplicate nodes for the same person. The challenge is to identify the duplicate nodes and reconstruct the family tree from these ambiguous views.

The continuous increase in privacy concerns by both individuals and enterprises is bound to increase the amount of data that is only available anonymously. In particular, the removal of personally identifiable information (PII) is becoming a common practice by both individuals and enterprises when releasing data into the public realm. Thus, a scenario where node labels are not available and neither any other information concerning nodes and edges is likely to become very important for different problems, including removing ambiguities.

Can synonyms be identified on an anonymous social network, when all PII has

been removed? This thesis focus on this problem and address different facets. In particular, we characterize and model the problem and propose algorithms to identify synonyms on anonymous social networks in the sense that node labels have no information (ex. randomly assigned integers). We will show that just the network structure encodes enough information to perform disambiguation in many cases.

1.1 Problem statement

In this section we formalize the network ambiguity problem. Consider a graph $G = (O, E)$ where the vertex set $O = \{o_1, \ldots, o_n\}$ represents objects and the edge set E represents pairwise relationships among the objects. Lets assume that objects have labels and, in particular, let $L_i = \{l_{i,1}, \ldots, l_{i,s_i}\}$ denote the set of labels that are assigned to object o_i, where s_i is the number of different labels that can be assigned to object O_i. Note that objects have one or more labels that are not necessarily unique. Thus, labels of different objects can be identical.

Consider an observation process of relationships among objects that reveals object labels. Thus, a relationship $(o_i, o_j) \in E$ is observed as (l_i, l_j) where $l_i \in L_i$ and $l_j \in L_j$. Let $L = \bigcup_{i=1}^{n} L_i$ denote the set of all different labels. The observation process applied to many (possibly all) relationships $(o_i, o_j) \in E$ will then yield a graph $G' = (L', E')$ where the vertex set $L' \subset L$ represents all observed labels and the edge set E' represents all observed relationships among labels. Note that a single label, $l \in L'$, can refer to one, two or more objects and different labels, $l_1, l_2 \in L'$, can refer to the same object.

The network disambiguation problem is to recover G (network of objects) having observed G' (network of labels) and this leads to two challenges. One is to identify when identical labels are used to refer to different objects and, thus, one vertex in G' could be mapped into two or more vertices in G. This is known as the homonym problem. The other challenge is to find different nodes that refer to the same object and, thus, merge two or more nodes of G' into one in G. This is known as the synonym problem. In this thesis we focus on the synonym problem and in this context we consider that labels of different objects are different, thus, $l_i \neq l_j$ for any $l_i \in L_i$ and $l_j \in L_j$ and for any $i \neq j$. Moreover, we assume there is no information on the labels themselves (i.e., labels are random numbers) and no information on the number of labels assigned to each object.

Figure 1.2 illustrates a network of labels and in Figure 1.3 the respective network of objects generated after solving the synonym problem. Note that the nodes $l_{1,1}$, $l_{1,2}$ and $l_{1,3}$ refer to the same object (o_1), while all other labels refer to their corresponding objects. Our goal is to study this problem in the context of social networks where nodes correspond to individuals and edges correspond to collaborations such

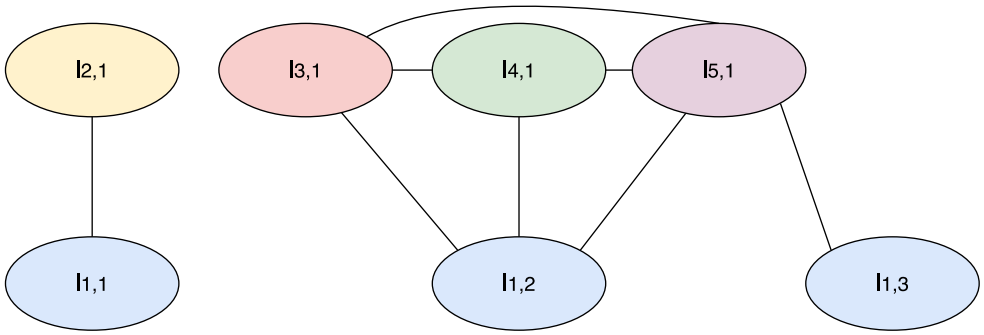


Figure 1.2: The network of labels G'. Note that the nodes $l_{1,1}$, $l_{1,2}$ and $l_{1,3}$ in G' all refer to the same object (o_1)

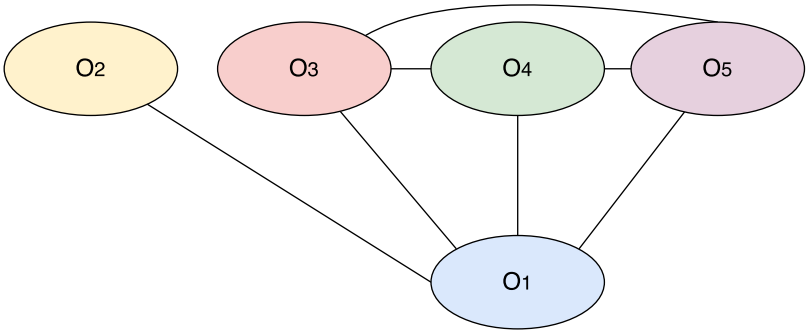


Figure 1.3: The network of objects, G, after solving the synonym problem.

as co-authorship of scientific papers.

1.2 Contributions

The main contributions of this thesis are listed bellow:

Characterization of name usage pattern

Consider the synonym problem where an object can be identified by different names. For example, consider a person and all its publications within her career. This person can use one name to sign her publications and then change her name and start use another name to sign her publications; or can sign alternating different names mixing them; or even can use one name the majority of the time and once use a different name, due to a mistake for example. We refer to these as name usage patterns.

Although most related works focus on solving name ambiguities, we first focus on classifying and characterizing multiple name usage pattern - the root cause for name ambiguity. By considering real examples of bibliographic datasets, extracted from DBLP and Google Scholar, we identify and classify patterns of multiple name usage by individuals, which can be interpreted as name change, rare name usage, and name co-appearance.

We propose a method to classify name usage patterns through a supervised classification task and show that different classes are robust (across datasets) and exhibit significantly different characteristics. In particular, for the usage of each individual we construct a time series for each of its names and then extract features from the time series in order to capture the individual's name usage pattern. These features are used in a data classification task to determine the type of pattern. A clustering algorithm separates the individuals by their name usage patterns.

We show that the collaboration network emerging around nodes corresponding to ambiguous names having different name usage patterns have strikingly different characteristics, such as their common neighborhood and degrees.

To the best of our knowledge, this is the first work that looks into the causes and types of ambiguity in synonyms and is described in Chapter 3. This work has lead to the following publication:

- Janaína Gomide, Hugo Kling and Daniel Figueiredo. Root causes for name ambiguity in scientific collaboration networks. In Scientometrics (2017) (journal).

Probabilistic model for name ambiguity in social networks

It is difficult to correctly characterize name ambiguity in real data. In the literature there are few studies that proposes mathematical models to study name ambiguity. For example, a model for homonym problem is to use the initials of the names leading to different persons having the same short name [4]. We are interested in the synonym problem and to the best of our knowledge, there is no mathematical model to characterize this phenomenon.

We propose a probabilistic model that introduces synonyms in a social network. The idea is to duplicate vertices and add/remove edges to neighbours of the original vertex. A duplicated vertex represents a second identification for the original vertex. Therefore, one object (vertex) of the original network can be represented by two vertices in the ambiguous network and relationships among the original object (vertex) can be copied to its duplicate and removed from the original.

The model has three intuitive parameters used for tuning the desired amount of ambiguity and can operate over any original social network. The proposed model has three phases, each with a parameter: (1) vertex duplication - with probability p a vertex is duplicated; (2) edge addition - with probability q an edge between a neighbour of the original vertex and the duplicated vertex is created; and (3) edge removal - with probability r an original edge that was copied to a duplicated vertex is removed. By varying these parameters, the model produces very different ambiguous networks. More details are in Chapter 4. This work was published in:

- Janaína Gomide, Hugo Kling and Daniel Figueiredo; A model for ambiguation and an algorithm for disambiguation in social networks. In CompleNet 2015, New York. 6th Workshop on Complex Networks (full paper).

Algorithms to identify synonyms in anonymous social networks

The growing privacy concerns among individuals and enterprises is leading to the removal of ever more personally identifiable information (PII) in data that is publicly available. However, classic solutions of name disambiguation heavily relies on contextual information with few recent exceptions, [4–6]. However all of them have considered the homonyms problem, where a single name could be used to identify multiple persons. We are not aware of any work that has specifically addressed the synonym problem in social networks when no PII is available.

We propose three different algorithms to solve the synonym problem using only the network structure. The algorithms tackle different aspects and scenarios: one uses heuristics based on the network structure to identify the duplicate nodes in the whole network; another one considers an ego-centered collaboration network and identifies all the vertices that represents the same individual; and the last algorithm is a framework that generate features based on subgraphs used as input to a neural network that classifies duplicate nodes. The three algorithms are described next.

The first is a simple and efficient algorithm that considers the entire synonymous network and identifies nodes that represent the same person. No label information is used but only structural characteristics such as distance, degree and common neighbours.

The algorithm leverages several structure-based heuristics to identify nodes in the network that represents the same person. For example, we consider that two nodes might refer to the same person if they are at distance 2, since it is unlikely that a node will have a relationship with itself using two different labels. Moreover, the same is considered if the common neighbourhood between two vertices strongly overlaps, and is contained in one another. We aim in developing a conservative approach to merge nodes, in order to minimize false-positives, allowing greater applicability of the algorithm. We apply this algorithm to ambiguous networks generated by the proposed probabilistic model. More details are in Chapter 4. This algorithm is published in:

- Janaína Gomide, Hugo Kling and Daniel Figueiredo; A model for ambiguation and an algorithm for disambiguation in social networks. In CompleNet 2015, New York. 6th Workshop on Complex Networks (full paper).

- Hugo Kling, Janaína Gomide and Daniel Figueiredo. A simple label-free algorithm for removing ambiguities in collaboration networks. In NetSci-x 2015,

Rio de Janeiro. International School and Conference on Network Science (poster).

The second algorithm that we propose considers the synonym ambiguity problem in the context of collaboration networks where individuals have a profile such as Google Scholar and DBLP. The goal is to consolidate the names appearing in the ego-centered collaboration network induced by the bibliographic records of profile owner.

Using just the network structure and no other information such as the node labels or metadata, we propose an algorithm based on the dominating set problem and the independent set problem that finds all nodes that correspond to the individual in its ambiguous ego-centered network. We apply this algorithm to real bibliographic data obtained from researchers' profiles. More details are in Chapter 5. This work has lead to the following publications:

- Janaína Gomide, Hugo Kling and Daniel Figueiredo. Consolidating identities of authors through egonet structure. In WebSci 2017, Troy. ACM on Web Science Conference (extended abstract).

- Janaína Gomide, Hugo Kling and Daniel Figueiredo. Consolidating author identities in anonymous ego-centered collaboration networks. Under preparation for submission (journal).

The third algorithm is an inductive learning framework to identify duplicate nodes in anonymous social networks. This novel framework is node centered and consists of first extracting induced and connected subgraphs from a graph rooted at the node. These subgraphs are matched to a reference graph to align the features across different instances of the same subgraph as well as to identify the equivalence classes within a subgraph. Finally, the features are aggregated across all extracted rooted subgraphs that are isomorphic, generating a single feature vector for each reference graph. The set of such vectors (across all reference graphs) is taken as the structural feature for the node.

These structural features are used as input to a two-level neural network designed to learn to identify duplicates. The first level learns to identify duplicates from the point of view of a single subgraph (and for every subgraph pattern). The second level integrates the classification of all subgraphs patterns to learn to identify duplicates.

We evaluate our framework on two kinds of social network: collaboration networks and family networks. In particular, the model trained in one network can effectively identify synonyms in another unseen network. More details are in Chapter 6. This work has lead to the following publication:

- Janaína Gomide, Daniel Figueiredo and Bruno Ribeiro. Inductive Learning for Duplicate Identification in Anonymous Social Networks. Under preparation for submission (journal).

The various results of this thesis strongly suggest that the network structure alone has enough information to identify and remove synonyms in many social networks. This is a fundamental finding given the importance of data consolidation and privacy concerns.

This text is structured in the following sequence. Chapter 2 presents the main results in literature concerning name ambiguity. Name usage pattern in the synonym problem is discussed in Chapter 3, in which we propose propose a method to identify and classify different patterns of multiple name usage. Chapter 4, presents a model to introduce synonyms in a collaboration network and a naive algorithm using just a few network features to identify duplicated nodes in the collaboration network. Chapter 5 presents our algorithm based on the dominating set and independent set problems to identify nodes of an ambiguous egonet that correspond to the egonet owner using only structural properties. Chapter 6 presents and evaluates an inductive learning framework to identify duplicate nodes in anonymous social networks. The conclusions and future perspectives are presented in Chapter 7.

Chapter 2

Related Work

The name ambiguity problem occurs when there is an uncertainly about the matching between names and objects. The uncertainty is caused because more than one name could have been applied to a single object (synonym) or the same name could have been assigned to different objects (homonym).

This problem has been subject of study from a variety of fields, including databases, machine learning, natural language processing and information retrieval. In [7] the different views of this problem are summarized as: (1) deduplication problem, to cluster different mentions to the same entity are clustered; (2) record linkage problem, to link records that match across databases and (3) reference matching problem, to match noisy records to clean records in a reference table. In this work we will focus on the deduplication problem in a social network where a single person can have different representations.

Name ambiguity observed in real datasets has been widely studied over the past decades in different contexts, such as person's names in web searchs [8, 9], name references in email archives [10], gene-protein names [11], geographic names [12] and author's names in digital libraries [2, 13].

There are some initiatives to solve this problem by manual inspection such as manual assignment by librarians [14] or collaborative efforts[1]. Another attempt is the use of an unique identifier for each author such as the one proposed by Open Researcher Contributor Identification[2] (ORCID), which provides a persistent digital identifier to uniquely identify the researchers and link them to their professional activities. Although interesting, these initiatives apply only for the context of digital libraries and require heavy human efforts, which prevent them from being used in massive name disambiguation tasks and in existing datasets that are already ambiguous.

A taxonomy concerning name ambiguity is proposed in [2] to organize the most

[1] `https://meta.wikimedia.org/wiki/WikiAuthors`
[2] `http://orcid.org`

representative automatic methods. According to this taxonomy, the methods may be classified following the type of exploited approach and also following the exploited evidences. The approaches are be author grouping [4, 10, 15–17], which tries to group the references to the same author using some type of similarity among reference attributes, or author assignment [18, 19], which aims at assigning the references to their respective authors.

Alternatively, the methods may be grouped according to the evidence explored in the disambiguation task: attributes of the citation, web information, or implicit data that can be extracted from the available information. The majority of methods [3, 16, 17, 19–26] uses at most the three main citation attributes: author names, work title and publication venue. Few methods [27, 28] exploit additional evidence such as emails, addresses, paper headers etc., which are not always available or easy to obtain. Some methods exploit relationships among authors and co-authors, usually represented as a graph [3, 4, 19, 23–25, 29, 30].

The majority of the methods use machine learning techniques to tackle the problem and most of them tackle the homonym problem. Supervised algorithms are used in [3, 4, 15, 16, 20, 31] and unsupervised algorithm (clustering) are used in [17, 21, 32]. Some methods receive the correct number of authors in the collection as input [19, 21, 22] or this number corresponds to the number of authors in the training data [20]. Other methods, such as those proposed in [16], try to estimate this number. In [33] the proposed algorithm mix clustering algorithm with human annotations to improve the results.

The problem of imbalanced training data has been treated in [34] where negative training data are often significantly larger than positive training data for the homonym problem. When we evaluate our algorithms for the synonym problem, most of the datasets are imbalanced and we analyze the metrics that are more appropriated for this scenario. Another issue is the scalability problem when disambiguating an entire repository. In [35] a new method, specially designed for the incremental scenario is proposed for the homonym problem. One of the algorithms proposed in our work, Chapter 6, shows that with a small percentage of network nodes it is possible to train a classifier to obtain a good performance for the synonym problem across the network. In that algorithm, each node is considered separately and as a new node requires just the extraction of its features (by looking a neighbourhood of distance at most five). It is not necessary to have access to the entire network.

A lot of the effort in name disambiguation is dedicated to the case where a single name is used by multiple individuals (homonym) [3, 4, 17, 19, 25, 30, 36]. Wang et al. [36] describe the homonyms problem as given a person with name a and a collection of N associated documents D^a the goal is to find how many distinct persons the

documents should belong to and to cluster the documents by the groups (persons) with high accuracy. In the perspective of networks, the homonym problem leads to a network where a node can represent multiple people. This scenario is prevalent in a context where people have few and short names, such as with Chinese or Japanese names, or when the names are abbreviated.

On the other hand, in a context where people have many and long names, such as in Brazil, ambiguities where a single individual appears with multiple names (synonym) is more prevalent. Also, when the network is constructed by multiple user views, such as in [1] where a familial network are created by different members that report theirs family relationships using different names to represent the same person. According to Kouki et al. [1] the problem is addressed using name similarity (based on string matching), relational information such as sibling overlap and logical constraint such as transitivity and probabilistic soft logic model. While in other works [1, 29] the synonym ambiguity problem is considered separately, and it has always been studied in the literature jointly with the homonym problem [15, 20–24, 27, 28, 37].

Another perspective of the ambiguity problem is approached in [38] and is the entity linking with a knowledge base. So, given a knowledge base containing a set of entities and a text collection in which a set of named entity mentions are identified in advance, the goal of entity linking is to map each textual entity mention to its corresponding entity the knowledge base.

This thesis focus on the ambiguity problem that arises when multiple names are used for one object (synonyms) in the context of social networks. Moreover, since networks are anonymous, the only available evidence is the relationships between the nodes and no other information is available. We show that just these relationships encode enough information in contrast to the works presented in survey [2].

Methods proposed in [1, 23, 24, 29] both consider the synonym problem and use network information. Despite using the network structure to solve the synonyms problem, these methods measure name similarity and leverage other contextual information, an approach very different from what this thesis considers.

2.1 Name ambiguity problem in anonymous networks

The growing privacy concerns is leading to the removal of even more personally identifiable information (PII) in data that is publicly available. With the removal of all PII, the social network becomes anonymized, and no contextual information is available (i.e., nodes have random labels).

Recently, a few works have considered the name disambiguation task in the context of privacy. In [4, 5] relational data is used and a collaboration network is built to solve name ambiguity using structure features. In [4] no context information is leveraged in the network, the authors characterize the similarity between two nodes based on their local neighborhood structure using graph kernels and solve the resulting classification task using SVM. The evaluation of their methodology used a proprietary dataset and the Internet Movie Database (IMDb) where they artificially introduced ambiguities (to have a ground truth).

In [5] a collaboration network is used in a method for solving entity disambiguation from timestamped link information obtained from the collaboration network. The method uses only the graph topology of an anonymized network and apply an unsupervised approach based Markov clustering. Experimental results are reported on two real-life academic collaboration networks, DBLP and Artminer.

In [6] a set of documents is partitioned according to name references and three different networks involving person to person, person to document and document to document are constructed and leveraged to solve name ambiguity without name information. The proposed method uses a novel representation learning model to embed each document in a low dimension vector space and identify ambiguities by hierarchical agglomerative clustering algorithm. To evaluate the algorithm they used the Arnetminer and CiteSeerX repositories and considered 10 highly ambiguous name references (homonym problem).

All these works that tackle the ambiguity problem considering anonymous network, [4–6], consider the homonym problem. Our work focus on the synonym problem using anonymous network.

Unlike most of the related work, in Chapter 3 we do not propose a mechanism to remove name ambiguities. The focus is to characterize the root causes for ambiguities (in the synonym problem) by studying multiple name usage patterns of individuals over time. By identifying classes of name usage patterns in bibliographic data and their consequences on the structure of the collaboration network, we contribute to building more effective name disambiguation algorithms. In Chapters 5 and 6 we propose algorithms that consider an anonymous network to the synonym problem. In Chapter 5 the algorithm finds duplicate nodes of the authors by solving a variation of the the dominating set and independent set problem specifically for the synonym problem, and excellent results are obtained in real ego-centered networks. Finally Chapter 6 proposes a framework to generate features based on rooted induced subgraphs and presents a two level neural network that uses these features to train and classify nodes as duplicate and results are excellent even when training with a small portion of the dataset and evaluating in networks never seen before during training.

Chapter 3

Name usage pattern in the synonym ambiguity problem in bibliographic data

Although various approaches have been proposed to tackle name disambiguation in collaboration networks, we believe that is necessary to take a step back to explore why and how individuals use multiple names thus leading to ambiguities. We believe that knowing the root causes of name ambiguity could help to design better solutions.

We focus on understanding and classifying multiple name usage patterns of individuals. By considering positive examples in real collaboration networks, we identify and classify patterns of multiple name usage by individuals, which can be interpreted as name change, rare name usage, and name co-appearance. In particular, we propose a method to discover patterns of name ambiguity as a supervised classification task and show that different classes are robust (across datasets) and exhibit significantly different properties. We show that the collaboration network structure emerging around nodes corresponding to ambiguous names from different root causes have strikingly different characteristics, such as their common neighborhood and degree evolution. We believe such differences in network structure and in name usage patterns with respect to different root causes can be leveraged to design more efficient name disambiguation algorithms.

Figure 3.1 shows the name usage pattern of three individuals as it appears in a real bibliographic dataset. The plots show the total number of publications appearing with the individual's two most frequently used names. The difference in the multiple name usage pattern of the individuals is striking. Individual in Fig. 3.1a consistently uses one name across many years with a second name being used in just one year (2009). In contrast, individual in Fig. 3.1b used one name for a long period of time and in a given year (1995) switched to using another name. Last, individual

in Fig. 3.1c used two different names interchangeably across all the years.

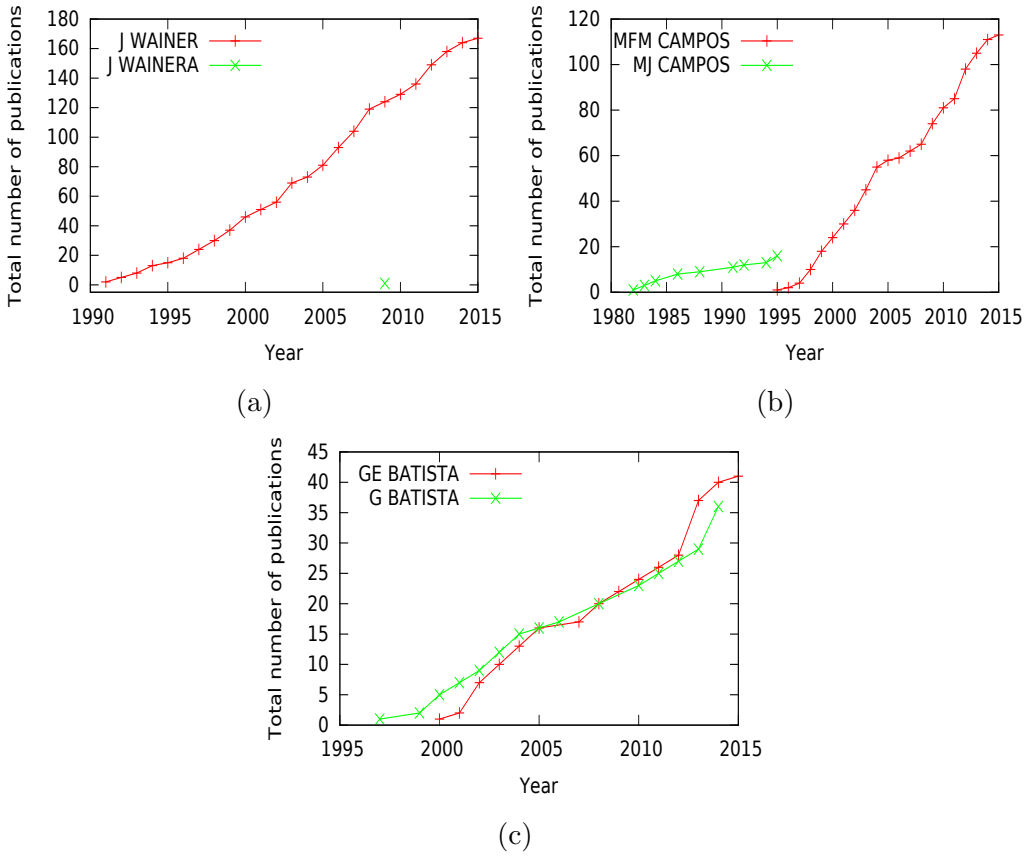


Figure 3.1: Different name usage patterns measured as the total number of publications with each name of a given individual over time: individual in Fig. 3.1a consistently uses a name with a second name only used in 2009; individual in Fig. 3.1b uses a single name until 1995 when changing to using another name; individual in Fig. 3.1c frequently and consistently uses two names along the years.

In this chapter, we address the following questions:

- Are these different name usage patterns representative and consistent across individuals suffering from the synonym problem?
- What are the consequences of these patterns to the structure of collaboration network?

As we will show, such patterns are indeed representative of the name usage for synonyms. Moreover, different patterns induce different fingerprints in the structure of the collaboration network. Although our focus here is not to remove ambiguity, our findings could be leveraged to design more effective algorithms. Last, to the best of our knowledge, this is the first work to classify name usage patterns of synonyms in real bibliographic data and quantify its effect on the collaboration network – a fundamental step to understand the origins of ambiguities emerging from synonyms.

By keeping with our objectives, we make the following contributions:

1. Propose a methodology to identify and classify different patterns of multiple name usage in bibliographic data.

2. Characterize the structural features in the collaboration network induced by different root causes for ambiguity.

The proposed method consists in building time series from the multiple name usage pattern of individuals and extracting features from them that are then used to classify the patterns using unsupervised and supervised classification tasks. Note that the methodology assumes knowledge of the multiple names of individuals since the goal here is not to identify such names but rather to unveil common patterns of name usage in real data and their consequences on the structure of the collaboration network. In particular, we show that a few very distinctive patterns are behind the majority multiple name usage in real data.

3.1 Synonym name patterns

The examples illustrated in Figure 3.1 suggest threes types of name usage pattern:

- *Rare*: When the individual uses one name the majority of the time and in rare cases uses the other name. The alternative name might have been misspelled or is the result of a typo.

- *Swap*: When the individual starts publishing with a name and some time later switches to another name. We consider that the individual in this case consciously changed the name used to sign publications.

- *Co-appearance*: When the individual has publications with both names in the same year or uses both names along the years it publishes. We consider that the individual has more than one form to write his name in publications.

Indeed, these three types are representative of patterns of multiple name usage found in real data, as we will soon demonstrate. Thus, we proposed a methodology to classify the name usage patterns of individuals that suffer from synonym into one of three classes. The idea is to map the multiple name appearances of an individual (across its publications in a bibliographic dataset) to a time series and use machine learning algorithms to classify the time series (one per individual) based on its features. Figure 3.2 exemplifies each step of the methodology that we describe next.

3.1.1 Duplicated name appearances

We start by assuming knowledge of a set of individuals and the names they have used within a given bibliographic dataset, for example, John von Neumann and its

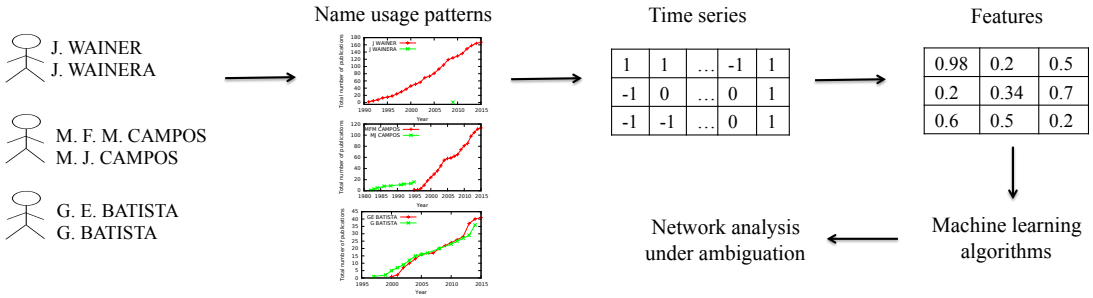


Figure 3.2: Steps of the proposed methodology to classify synonym patterns: multiple name appearances of an individual, time series representation, features extracted from the time series, classification of patterns using machine learning, and network analysis of different classes.

two names "J Neumann" and "JV Neumann" as they appear in Google Scholar. For each individual, we collect the corresponding set of publication records that include date of publication and names of co-authors.

In particular, let $U = \{u_1, u_2, u_3, ..., u_n\}$ denote the set of individuals with more than one name and let l_{i1}, l_{i2}, $i = 1, ..., n$ denote the two most frequently used names for individual u_i in the dataset. We limit the analysis to two names per individual because this is by far the most common case: in the official synonym data for DBLP, 95.8% of individuals with multiple names appear with exactly two names. Information concerning the ground-truth on synonyms for DBLP is described in Section 3.2.1.

3.1.2 Time series representation

For each individual we construct a time series for each of its names with the number of publications that have appeared with the given name per year. Thus, the time series has a yearly time scale. Let $T_i^{l_1}$ and $T_i^{l_2}$ denote the time series of individual u_i associated with its two names l_{i1} and l_{i2}. We provide empirical support for choosing just two names (the two most used) in Section 3.2.

We combine the multiple time series of an individual into just one time series as follows: for each year, if either label l_{i1} or l_{i2} is used then the new time series is assigned the value -1 or 1, respectively, for that year. If both labels are used in the same year then the new time series is assigned the value 0 for the respective year. Let T_i denote the combined time series for individual u_i.

To illustrate the construction of the time series, considers the individual shown in Figure 3.1(c). He has published using the name "G. Batista" from 1997 to 2014 and in 2000 he started to use the name "G. E. Batista" until 2015. Table 3.1 shows the time series representation for this individual. Note that 1998 does not appear in the time series since no publications appeared in the dataset for that year.

Table 3.1: Time series representation: example of how to combine the multiple time series of an individual. A value of 1, -1 or 0 is assigned to the combined time series if just label l_{i1} or l_{i2} or both are used in respective year.

Name	97	99	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
G. E. Batista			X	X	X	X	X	X		X	X	X	X	X	X	X	X	X
G. Batista	X	X	X	X	X	X	X	X	X		X		X	X	X	X	X	
Result	−1	−1	0	0	0	0	0	0	−1	1	0	1	0	0	0	0	0	1

3.1.3 Time series features

We extract features from the time series in order to capture the individual's name usage pattern. In particular, since we would like to classify individuals into one of the above three classes, features of the time series should reflect these classes. Thus, the first insight is related to the number of changes that occur in the time series. Is name change frequent or rare for a given individual? The second insight is related to the number of times that both names are used. Does the individual really has two ways of writing his name? The third insight is related to the appearance of the least used name. Is this frequently or rarely used with respect to the most used name?

In order to capture these insights we introduce three features that will be extracted from the time series. Let N_i be the length of time series T_i defined by the number of years with a -1, 0 or 1 entry, and $N_{i,0}$, $N_{i,1}$ and $N_{i,-1}$ the number of zeros (both names), ones (only l_{i1}) and minus ones (only l_{i2}) in the T_i, respectively. Let $T_{i,j}$ denote the j-th element of the time series of individual u_i. Let $S_i = \sum_{j=2}^{N} \mathbb{1}(T_{i,j} \neq T_{i,j-1})$ denote the number of changes with respect to the previous year of user u_i and $X_i = \sum_{j=2}^{N} \mathbb{1}(T_{i,j}T_{i,j-1} = -1)$ denote the number name alternations for user u_i. Note that X_i does not count when both names are used in the same year ($T_{i,j} = 0$). The features are defined as follows, noting that all of them are used in the range [0,1]:

- *Persistence (P_i)*: fraction of time that the same name appears in the following years:

$$P_i = 1 - \frac{S_i}{N_i - 1} \tag{3.1}$$

- *Co-appearance (C_i)*: fraction of time both names appear in a year combined with fraction of name swaps:

$$C_i = \frac{N_{i,0} + X_i}{N_i} \tag{3.2}$$

- *Rare (R_i)*: fraction of time the least frequently used name does not appear

alone:

$$R_i = 1 - \frac{\min(N_{i,1}, N_{i,-1})}{N_i} \tag{3.3}$$

3.1.4 Data classification

Can individuals be correctly classified based on their features by an unsupervised classification algorithm? To answer this question, we create a small training set by selecting randomly selecting time series and manually labeling them with one of the three classes (by inspection of the time series). Next, we run a clustering algorithm that takes as input just the features (i.e., three values) of the time series in this training set (and not the labels). If the features are an adequate discrimination of the data, then the clustering algorithm should classify the data in accordance to the manual labels.

In this work, we adopt the simple k-means clustering algorithm [39] using the Euclidean distance between points (with $k = 3$, since we have three classes). To assess the quality of the clustering produced by the algorithm, we consider the following metrics: homogeneity – measures if clusters contain only members of a given class; completeness – measures if all members of a given class are assigned to the same cluster; and Adjusted Rand Index (ARI) – measures the similarity between pairs of clusters.

Next, we train a model to learn to classify the time series into one of the three classes. We use the same training set where each individual has three features (the three proposed metrics) and the label (manually assigned by inspection of the time series). This trained model is then used to classify all individuals in our dataset (using as input the corresponding feature values), allowing us to study the characteristics of different classes with a large dataset, such as the impact of the class in the collaboration network.

For this supervised learning, we adopt the Support Vector Machine (SVM) algorithm [40]. An SVM model is a representation of the training data as points in space, projected such that data from different classes are as far as possible from one another. In particular, the algorithm searches for an optimal hyperplane which separates the training data. A large fraction of the training set is used to create the model, while the complement fraction is used to verify the quality of the trained model.

3.1.5 Network analysis under ambiguity

It has recently been observed that structural features of collaboration networks depend and are crucial in tackling both kinds of ambiguity problems [3, 24, 41–43]. These networks are usually constructed from datasets by considering each record

(e.g., paper or book) separately. In particular, each record has a set of names (e.g., co-authors of the paper) that are mapped to nodes and form a clique in the collaboration network. Thus, the network is formed by the union of cliques, one per record, with nodes identified by the names appearing in the record. Clearly, individuals subject to synonyms will appear as multiple nodes in the network, giving rise to a misleading network structure [41, 43, 44].

To measure the impact of different causes of ambiguity in the collaboration network we consider the following structural characteristics concerning nodes (names) of a given individual:

- Node degree: we will compare the degree of the two nodes (names) for the different types of ambiguity;
- Number of common neighbors: the size of the common neighborhood of the two nodes (names);
- Distance: the distance between the two nodes (names) in the network;
- Degree evolution: the derivative with respect to time of the degree growth for both nodes (names). By looking to this metric we want to verify how is the growth of the degree of an node through the time and note if it has not increased or if it is constantly growing, for example. Let us define the first (t_i^{min}) and last (t_i^{max}) year of publications of a given author u_i as follows:

$$\begin{aligned} t_i^{min} &= \min(\min(T_i^{l_1}), \min(T_i^{l_2})) \\ t_i^{max} &= \max(\max(T_i^{l_1}), \max(T_i^{l_2})) \end{aligned} \tag{3.4}$$

  Let the degree variation of an author's label be the difference in number of neighbors in the network between it's first and last publication year. Finally, degree evolution for a label ($e_i^{l_{1,2}}$) is defined as the ratio between degree variation and period of publications as follows:

$$e_i^{l_{1,2}} = \frac{d_i^{l_{1,2}}(\max(T_i^{l_{1,2}})) - d_i^{l_{1,2}}(\min(T_i^{l_{1,2}}))}{\min(1, t_i^{max} - t_i^{min})} \tag{3.5}$$

3.2 Empirical evaluation

In this section we describe the datasets and the results obtained by applying the proposed method.

3.2.1 Datasets

The dataset used to evaluate the proposed method can be any collections of records of collaboration with positive instances of individuals that appear in the data with multiple names. In order to study the patterns of name usage in the synonym ambiguity problem, it is necessary to know the names used by a given individual in a given dataset.

We used two different real bibliographic records that are used to build collaboration networks and are readily available online: the DBLP (Digital Bibliographic Library Project)[1] and a small subset of Google Scholar[2] that we collected.

DBLP

The DBLP contains bibliographic information for publications in Computer Science and it has been widely used in many studies. In particular, we considered the entire DBLP dataset and the version of May 2014, which can be freely downloaded in XML format[3]. In this repository, some individuals do appear with different names in distinct publication records, giving rise to the synonym problem. However, DBLP maintains a record for a known set of authors that appear with multiple names in the database, that are identified in a different section of the same XML file.

In particular, some elements of the XML file have a tag `<www>` and attribute `key="homepages..."`. These elements represent the homepages of individuals within the DBLP website and include a list of names used in the publications of the corresponding individual. To illustrate, the example below was extracted from the XML file. The first tag `<www>` is for the homepage `homepages/75/6127` which corresponds to the individual R. Alberdi. The second tag `<www>` is for the homepage `homepages/75/3969` which corresponds to an individual that uses two names in its publications: Yusuke Mitari and Yusuke Mitarai, as indicated by the tag `<author>`.

```
 ...
<www mdate="2009-06-10" key="homepages/75/6127">
<author>R. Alberdi</author>
<title>Home Page</title>
</www>
<www mdate="2012-09-26" key="homepages/75/3969">
<author>Yusuke Mitari</author>
<author>Yusuke Mitarai</author>
```

[1] http://dblp.uni-trier.de
[2] http://scholar.google.com
[3] Link to download the XML of the entire DBLP database: http://dblp.uni-trier.de/xml/

```
<title>Home Page</title>
</www>
...
```

Thus, DBLP has identified some authors that use more than one name in their publications, as well as the names used by these authors[4]. Therefore, we use only this set of individuals and their corresponding names as our ground truth for synonyms in DBLP not performing any sort of data pre-prossessing. Last, the DBLP dataset used in this work listed 14,290 individuals with more than one name and a total of 29,126 different names among them, although most of them have just two names (see Table 3.2).

Google Scholar

Google Scholar provides a wide range of bibliographic records from the academic publications. It draws on information from university repositories, publishers and websites that were identified as scholarly. Google Scholar maintains a profile for each individual that lists all publications of the person along with other information, such as full name and affiliation. After approval by the owner, the profile becomes publicly accessible.

The name appearing in the author's profile is not the same name that appears in the list of publications of the profile. In particular, Google Scholar shows only the initials and the last name of the authors of the publication. In spite of this simplification, the owner of the profile still appears with different names in the list of publications, giving rise to the synonym problem.

In order to select profiles in Google Scholar, we considered individuals that were awarded a research fellowships from CNPq[5] in six different areas (Biophysics, Computer Science, Electric Engineering, Philosophy, Medicine and Sociology). We chose this target group since Brazilians tend to have many first and last names, and use them rather freely in publications.

From the 1629 researchers that receive a research fellowship from CNPq, 1060 have a public Google Scholar profile (we automatically search for their public profiles on Google Scholar using their names). We collected all publications available in the profile for each individual, and found 881 profiles where the listed publications exhibit more than one name for the profile owner. We mapped 3408 different names used to refer to these 881 individuals.

[4]See details on DBLP's handling of synonyms at http://dblp.uni-trier.de/faq/How+does+dblp+handle+homonyms+and+synonyms.html

[5]CNPq is the Brazilian National Research Council responsible for funding research, similar to the National Science Foundation (NSF) in the United States of America.

Note that Google Scholar does not have a ground truth, meaning that it does not mark which name in a given publication corresponds to the profile owner. However, since we know the name of the profile owner and the name appearing in the profile heading, we search within the author names of the publication for the name that most likely corresponds to the profile owner (using text-based heuristics and string distances). Thus, for each publication, we determine the corresponding name for the profile owner.

In order to measure the quality of the data collected, we should verify if the profile is really from the person we are expecting, if the profile owner appears in the publications listed in the profile, and if the name we identified in the publication corresponds to the profile owner. We randomly selected 20 Google Scholar profiles and manually verified these issues. We found that all profiles do correspond to the expected individual. Among all publications listed for 19 individuals (a total of 1679), 6% of them do not contain the name of the profile owner. However, these publications are discarded and have no impact in our analysis. One individual had 1042 publications listed but his name did not appear in 388 of them (they seem to belong to a different person), and thus these publications were discarded. Last, in all cases that a name was identified within the co-authors of a publication, the heuristic did find the profile owner.

3.2.2 Time series and number of names

After creating the datasets we built the time series for each individual for its two most used names ($T_i^{l_1}$ and $T_i^{l_2}$) and combined them into a single time series T_i. Then, for each T_i the features P_i, C_i and R_i were computed, as described in Section 3.1.

Although our methodology is limited to two names per individual, most individuals in the dataset here considered tend to appear with just two names or tend to use more frequently just two names. Table 3.2 shows the number of the names used by an individual in its publications in the datasets. Note that in the DBLP ground truth, the vast majority of individuals use just two names (96%), while in the profiles collected in Google Scholar, this value is much lower (24%). However, name usage frequency is far from uniform, meaning that an individual can appear with many names across all its publications, but mostly use one or two names.

Without loss of generality, let $|T_i^{l_j}|$ denote the number of times that the j-th most used name of an individual i has appeared. Thus, the ratio $|T_i^{l_j}|/|T_i^{l_1}| \leq 1$ denotes the relative frequency of the j-th most used name with respect to the most used name. Figure 3.3 shows the average and standard deviation for this ratio (for different values of j) across all individuals (of both datasets) with more than one name. Note that the third most used name appears (on average) less than 10% of

Table 3.2: Number of names per individual in the DBLP ground truth and in the collected Google Scholar profiles.

Number of names	DBLP ground-truth	Google Scholar
1	0 (0%)	179 (16.9%)
2	13700 (95.9%)	256 (24.2%)
3	569 (4.0%)	201 (19.0%)
4	21 (0.2%)	121 (11.4%)
5 or more	0 (0%)	303 (28.6%)
Total	14290	1060

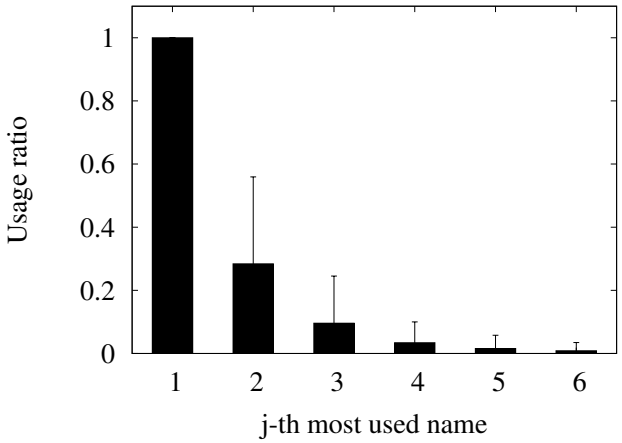


Figure 3.3: Average and standard deviation of the relative ratio ($|T_i^{l_j}|/|T_i^{l_1}|$, across all i) of the number of appearances of the j-th most used name.

the time with respect to the most used name. The name usage frequency decreases very fast for less frequent names. Thus, considering just the two most frequently used names, they account for the vast majority of the individuals in DBLP or the majority of name appearances in Google Scholar.

We discard individuals with too few name appearances in order to avoid outliers, keeping just individuals with at least 10 name appearances among its two most used names ($|T_i^{l_1}|+|T_i^{l_2}| \geq 10$). After this filter, there were 5497 individuals with multiple names in DBLP and 608 individuals in Google Scholar.

3.2.3 Data classification

We selected 200 random individuals from the dataset and manually labelled their time series (by inspection) into one of the three classes. This labelled data will be used to assess the quality of the clustering algorithm that produces clusters using just the data features (persistence, co-appearance and rare). The labelled data will also be used to train and assess the quality of the classification algorithm.

We run the k-means algorithm ($k = 3$) to cluster the training dataset based on its features (P_i, C_i and R_i). Figure 3.4 shows a 2D projection of the points

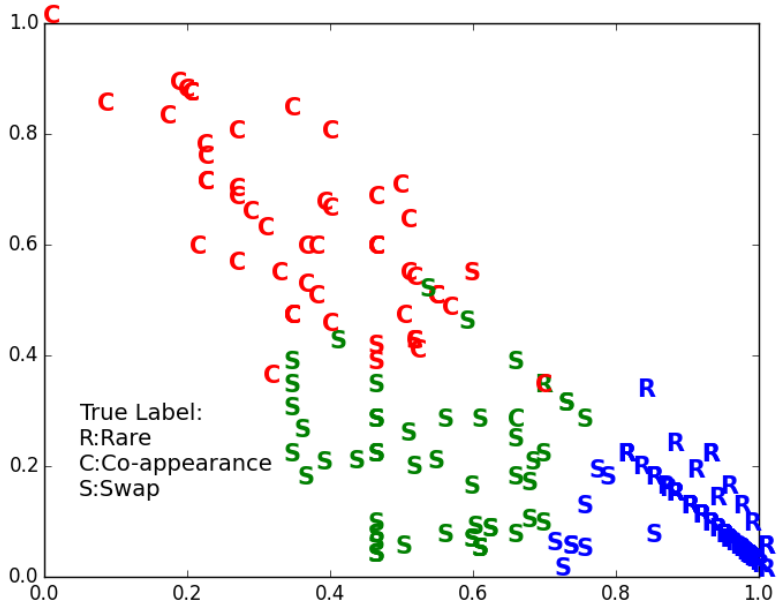


Figure 3.4: Clusters generated based on the features of the individuals' time series. The letters represents the true label (manually determined by inspection) and the colors represent the cluster assigned by the clustering (unsupervised) algorithm. The x axis represents the value of the feature R and the y axis represents the value of the feature C.

corresponding to the different individuals (x-axis and y-axis correspond to the values for features R and C, respectively). The letters represent the manual labels given to each data point (individual) and the colors represent the cluster of each individual assigned by the algorithm. Note that there is very good correspondence between the manual (true) label and the assigned cluster. This indicates that the features can be used to classify the individuals into their corresponding class.

Table 3.3 shows five examples of individuals from each cluster. Note that individuals in the red cluster have used both names most of the time (many 0 entries in their time series) or changes the name very often and thus characterizes the Co-appearance class. Individuals in the green cluster have time series with mostly just 1 or just -1, meaning that these individuals use one name most of the time and rarely use the other name. This behavior characterizes the Rare class. Finally, individuals in blue cluster have time series that start with -1 but switch to 1 after some time, meaning that in the beginning one name was used and later another one. This behavior is expected for the Swap class.

Table 3.4 presents the three metrics to measure the quality of the cluster produced by the algorithm, taking as ground truth the manual labels. Note that both homogeneity and completeness is larger than 75% (in a scale from 0 to 1), meaning that the clusters are quite homogeneous and complete. Moreover, the Adjusted

Table 3.3: Peaking into the clusters: feature values and time series of the individuals in each cluster. The color also corresponds to Figure 3.4.

Cluster	Features			Time series
	R	C	P	
Red	0.4000	0.6000	0.6429	-1 1 0 0 0 0 0 -1 -1 0 0 0 -1 -1 -1
	0.3077	0.6154	0.4167	-1 0 0 0 0 -1 1 1 0 0 1 0 1
	0.1111	0.8333	0.7647	-1 0 0 -1 0 0 0 0 0 0 0 0 0 0 0 0 1
	0.2727	0.7273	0.7000	-1 -1 0 0 0 0 -1 0 0 0 0
	0.5000	0.5000	0.6364	-1 0 0 0 -1 -1 0 0 0 -1 -1 -1
Green	0.8000	0.2000	0.7778	-1 -1 -1 -1 -1 -1 -1 0 0 -1
	0.9091	0.0909	0.8000	-1 -1 -1 -1 -1 -1 -1 -1 -1 0 -1
	0.9167	0.0833	0.9091	-1 1 1 1 1 1 1 1 1 1 1 1
	0.9333	0.0667	0.8571	-1 -1 -1 -1 0 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1
	0.9231	0.0769	0.9167	-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1
Blue	0.5000	0.1667	0.6364	-1 -1 -1 0 -1 0 1 1 1 1 1 1
	0.6364	0.0909	0.9000	-1 -1 -1 -1 1 1 1 1 1 1 1
	0.6364	0.0909	0.9000	-1 -1 -1 -1 -1 -1 -1 1 1 1 1
	0.5714	0.2143	0.7692	-1 -1 -1 -1 -1 -1 -1 0 0 -1 1 1 1 1
	0.7857	0.0714	0.9231	-1 -1 -1 1 1 1 1 1 1 1 1 1 1 1

Table 3.4: Measuring the quality of the (unsupervised) clustering using the manual labels as ground truth.

Homogeneity	Completeness	Adjusted Rand Index
0.751	0.759	0.788

Rand Index (ARI) value is 78% (in a scale from -1 to 1, with 0 being random), meaning that the clusters found by the algorithm are almost perfect. Thus, we conclude that the proposed features can indeed be used to classify individuals into their corresponding name usage class.

We proceed to classify all individuals from both datasets into their corresponding class. In order to accomplish this, we consider the 200 manually labeled individuals and use their features (P_i, C_i and R_i) to create a training set with examples from all the three classes. We next create a model with this training set using the SVM algorithm and, in order to assess the robustness of the learned model, we measure the performance of the 10-fold cross validation (90% of training set used to training the model, 10% to validate it) and the performance when varying the training set size. We repeat the 10-fold cross validation 1,000 times and report on the average result across these independent runs (usually much more accurate than a single run).

The results obtained by the 10-fold cross validation are shown in Table 3.5. Note that the error inside and outside the training set is very low, and that the precision and recall achieved by the classifier (on the remainder of the training set) is around 96% indicating that the learned model is robust and that classification was very

Table 3.5: Measuring the quality of the training set using 10-fold cross validation.

E_{in}	E_{out}	Precision	Recall
0.0319 (0.0008)	0.0340 (0.0030)	0.9727 (0.0028)	0.9660 (0.0030)

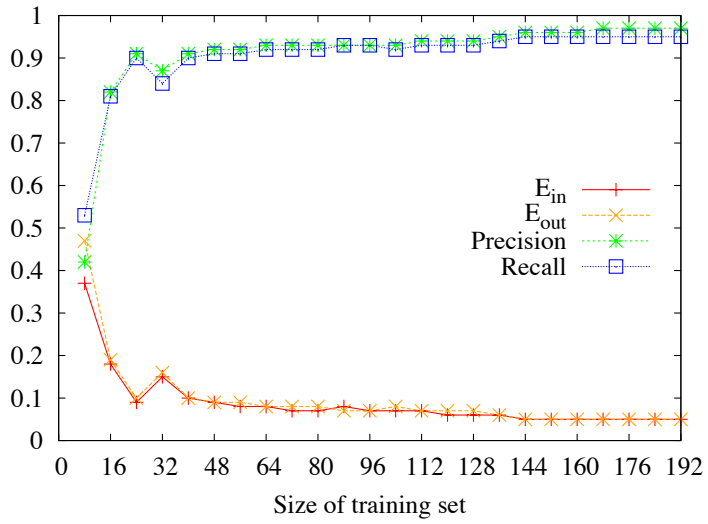


Figure 3.5: Performance of the classification as a function of the training set size.

accurate.

We now consider different training set sizes (chosen uniformly at random from the full training set) to train the classifier, and measure the same performance metrics (error inside and outside the training set, precision and recall). Results are shown in Figure 3.5, as a function of training set size. Note that with just forty examples we can achieve low error in and outside the training dataset and good precision and recall. As the training dataset grows the precision and recall are both higher than 90%. Thus, the size of (full) training set manually annotated is adequate.

After verifying the excellent performance of the classifier on the training set, we used the trained SVM model to classify all the individuals in both datasets. The distribution of the individuals among the three classes is shown in Table 3.6. Around half of the individuals were placed in the Rare class in both datasets, indicating that this is likely to be the main cause for synonym ambiguity, probably due to a rare name variation (i.e., using the middle initial) or even name misspelling. In DBLP, the percentage of individuals that change names (Swap class) and that use both names (Co-appearance class) are approximately the same, while in Google Scholar there is a smaller percentage of individuals classified in the Swap class. This difference might be caused by the preprocessing done by Google Scholar to the names of authors as found in publications, in a explicit attempt to reduce synonyms.

Table 3.6: Percentage and number of individuals from the DBLP and Google Scholar classified into each synonym ambiguity class.

Classes	DBLP	Google Scholar
Rare	40.82% (2244)	49.84% (303)
Swap	33.78% (1857)	17.93% (109)
Co-appearance	25.40% (1396)	32.23% (196)

Table 3.7: Average and standard deviation of feature values for each class. Note the distinctive combination of values of the various classes.

Classes	P_i	C_i	R_i
Rare	0.81(0.11)	0.12(0.07)	0.88(0.06)
Swap	0.74(0.11)	0.25(0.10)	0.63(0.10)
Co-appearance	0.50(0.15)	0.49(0.13)	0.46(0.14)

The average value within each class of the features used for classification are shown in Table 3.7, illustrating the differences between the classes. Notice that feature C_i has the largest average value for individuals classified as Co-appearance indicating that individuals in this class most of time publish with both names. For the class Rare, R_i and P_i have a high average value, indicating that individuals in this class have a name that appears very seldom and that one name persists for most of the time. For the class Swap P_i has the largest average value indicating that there are not many swaps between names, R_i is moderate indicating that there is not a name that appears only seldom, and the low value for C_i indicates that both names are not common for the individual.

3.2.4 Network under ambiguity

We now assess the impact that synonyms have on the structure of the collaboration network. We consider a collaboration network constructed by the superposition of all publications in found in our datasets, and not just the individuals that have been identified with multiple names. Moreover, the synonym ambiguity found in our ground truth was *not* removed, so each name appearing in the dataset corresponds to a node in the network. This allows to measure statistics and compare the different nodes that correspond to the same individual.

Table 3.8 shows some statistics of the networks constructed for both DBLP and Google Scholar datasets. Note that DBLP is a much larger network since we consider the entire database and we only have a rather small sample collected from Google Scholar (about 10 times smaller in number of nodes than DBLP, but with comparable average degree).

Figures 3.6 and 3.7 show the distribution for the degree and number of common neighbors for the DBLP and Google Scholar datasets. The degree distribution for

Table 3.8: Date and size of datasets and details of corresponding collaboration network.

	DBLP	Google Scholar
Date collected	May, 2014	August, 2015
Publications	2,553,549	154,012
Nodes	1,397,510	153,112
Edges	5,450,815	705,394
Avg. deg. (std)	7.8 (16.3)	9.2 (22.4)
Max. deg.	1,638	3,172

both primary and secondary vertices for both datasets (Figures 3.6a and 3.7a) exhibits a heavy tail behavior, following suit with the overall degree distribution of the network (not shown). As expected, the distribution for the primary vertex has a longer tail since its definition implies it has appeared in more publications. As indicated by the distributions, the average degree of primary nodes is much larger than that of secondary nodes, as shown in Table 3.9. Moreover, as with the most heavy tail distributions, the standard deviation is also quite high, in the order of the average, as shown in Table 3.9. Note that Google Scholar has a much larger average primary degree than DBLP, possibly because individuals in Google Scholar are recipients of a research fellowship which means they tend to publish more and thus have more collaborators.

The degree distribution of primary vertices conditioned on each class are relatively similar (not shown), while being quite distinctive for secondary vertices, as shown in Figures 3.6b and 3.7b. In particular the tail for the Rare class is much shorter than for the other classes, while Co-appearance exhibit the heaviest tail. Observe in Table 3.9 that the average degree of the secondary vertex of class Rare is the lowest in both datasets. This is because the individuals of this class seldom use its second name, while individuals of the class Co-appearance use their second name more frequently and, thus, have the highest average value of degree, in both datasets.

Figures 3.6c and 3.7c show the distribution of the number of common neighbors between the primary and secondary nodes, illustrating that this statistic is also class dependent and consistent across the two datasets (averages shown in Table 3.9). Again, we observe that the distributions exhibit a heavy tail, with the classes Rare and Swap having a shorter tail than Co-appearance, for both datasets. For class Rare this is intuitive as the second name is rarely used and thus tends to have a much lower degree, and consequently a lower number of common neighbors. For the class Swap, a possibility is that the two names (primary and secondary) are used in different contexts (e.g., areas of research) and thus have fewer common collaborators. In contrast, the two names of individuals in class Co-appearance have

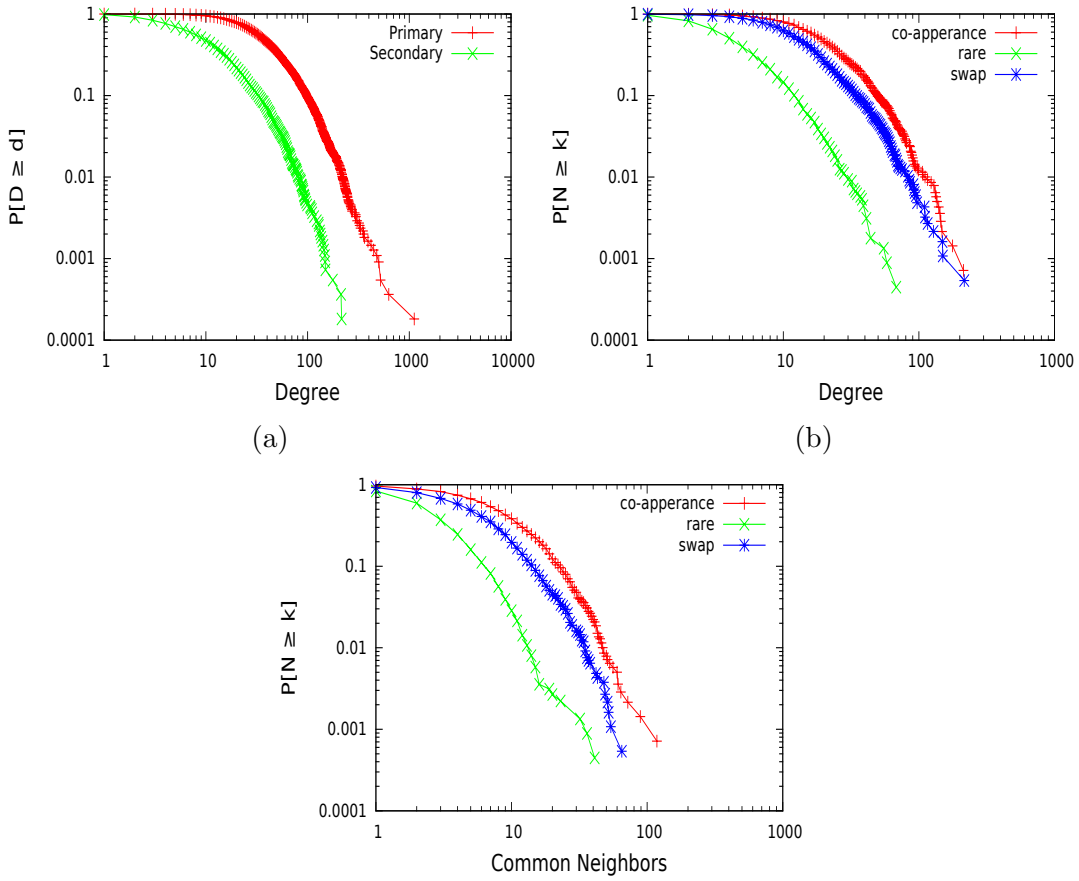


Figure 3.6: Degree and common neighbors distributions for DBLP. In Figure 3.6a degree distribution of the primary and secondary vertices (names) across all classes and in Figure 3.6b for each class for the secondary vertex. In Figure 3.6c distribution of common neighbors between primary and secondary vertices (names), per class.

the largest number of common collaborators, indicating that both names indeed are used interchangeably with the same set of collaborators.

We also consider the distance between the two nodes that represent the same individual in the network, with results shown in Table 3.10. Note that in both datasets the vast majority are at distance two, meaning that the two vertices (primary and secondary) have at least one common collaborator. Also note that some pairs lie in different connected components having no distance between them. Curiously, there are a few pairs that are at distance one (thus, they are neighbors), which implies that the two names must have appeared in the same publication. This can be due to having a collaborator (co-author in a paper) with a name that is identical to either the primary or secondary names, or to errors in preprocessing of the data by the curators (DBLP and Google Scholar). In any case, this is a very small percentage of cases and does not significantly impacts our analysis.

The last analysis considers the degree evolution for both nodes (names) during

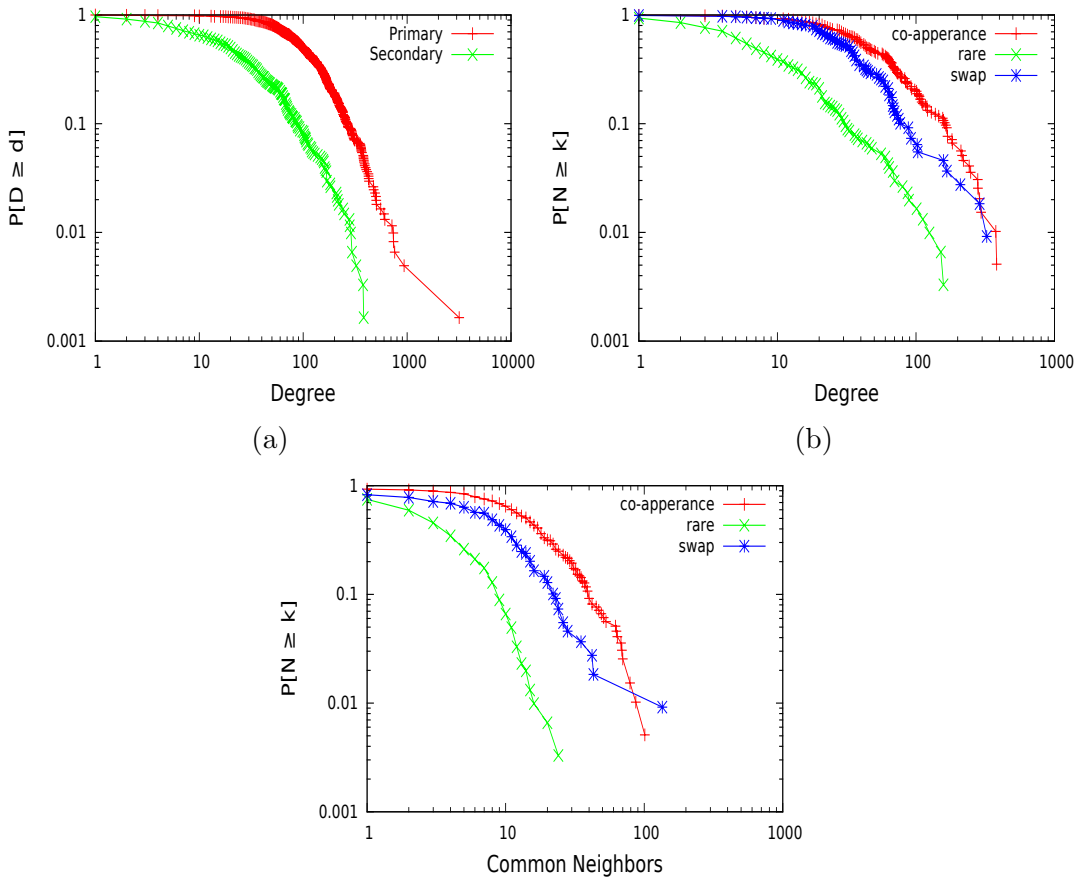


Figure 3.7: Degree and common neighbors distributions for Google Scholar. In Figure 3.7a degree distribution of the primary and secondary vertices (names) across all classes and in Figure 3.7b for each class for the secondary vertex. In Figure 3.7c distribution of common neighbors between primary and secondary vertices (names), per class.

Table 3.9: Average and standard deviation of degree and number of common neighbors (between primary and secondary vertices).

	DBLP	Google Scholar
Degree		
Primary	48.71 (46.43)	138.46 (172.45)
Secondary	14.07 (16.85)	36.40 (51.65)
Degree of secondary		
Rare	5.32 (5.78)	13.81 (21.27)
Swap	16.83 (16.37)	43.85 (48.75)
Co-appearance	24.46 (21.59)	67.17 (66.98)
Common neighbors		
Rare	2.62 (2.90)	3.26 (3.65)
Swap	6.28 (6.81)	10.01 (14.82)
Co-appearance	10.05 (10.17)	18.26 (17.45)

Table 3.10: Percentage and number of individuals whose corresponding nodes (primary and secondary names) are at distance one, two, three, four or more, or not connected in the collaboration network.

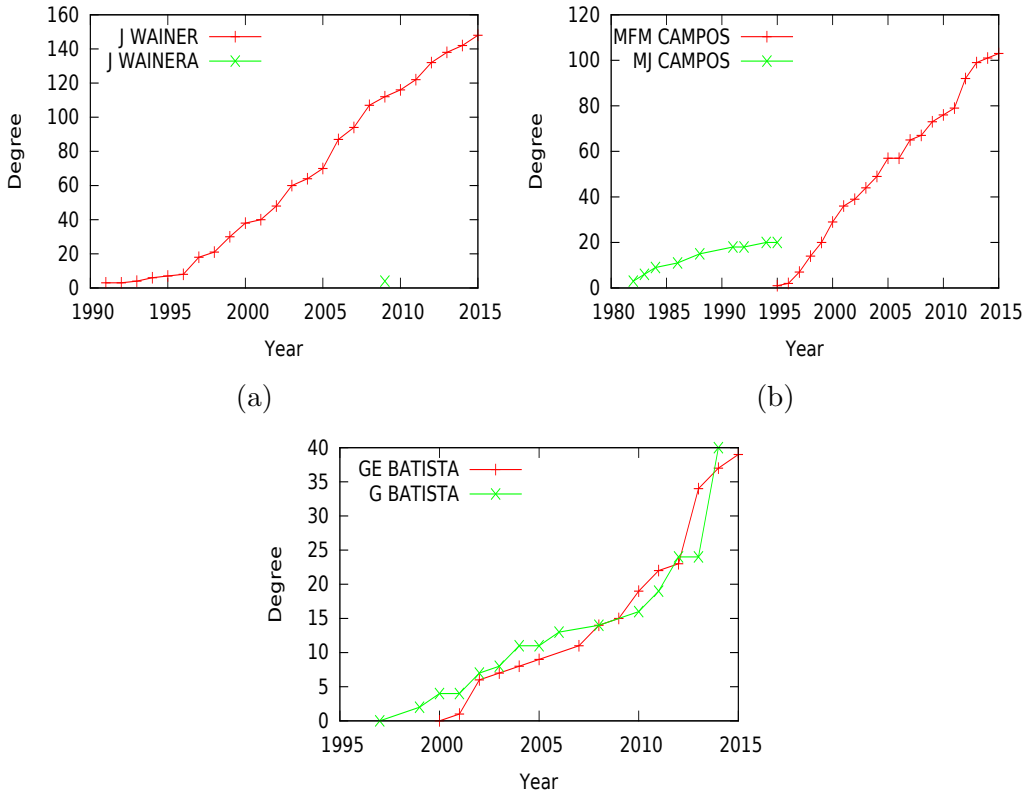
Distance	DBLP	Google Scholar
1	0.07% (4)	1.15% (7)
2	89.79% (4936)	80.92% (492)
3	2.83% (155)	4.12% (25)
≥ 4	4.44% (244)	5.92% (36)
∞	2.87% (158)	7.89% (48)

Table 3.11: Average and standard deviation of the derivative of the degree evolution for primary and secondary nodes (names) for each class in each dataset.

Dataset	Class	Primary	Secondary
DBLP	Rare	2.84 (2.75)	0.12 (0.24)
	Swap	2.25 (2.03)	0.77 (0.83)
	Co-appearance	2.47 (2.15)	1.25 (1.14)
Google Scholar	Rare	4.79 (3.70)	0.38 (0.62)
	Swap	5.21 (3.52)	1.48 (1.42)
	Co-appearance	4.91 (7.00)	2.17 (1.95)

the period of time that the individual has published. Figure 3.8 shows the degree evolution for each name of the three individuals of Figure 3.1. Interestingly, these two figures are very similar, indicating that publishing and building collaborations are strongly related. Note that the degree evolution of these three individuals are quite different and, thus reflects the difference between the three classes (Rare, Swap and Co-appearance).

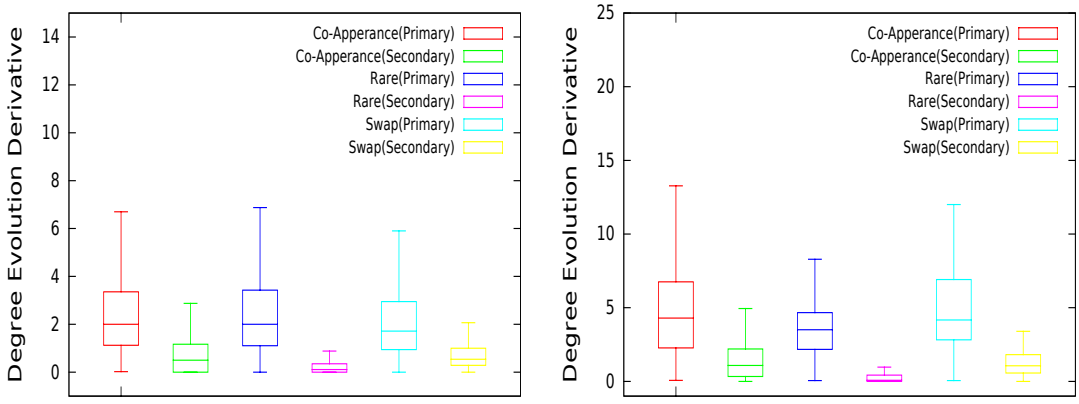
Figure 3.9 shows a boxplot of the derivative with respect to time of the degree evolution for both nodes (names) for the three classes in both datasets. Note that the for all classes, and in both datasets, the degree evolution for the primary node is much larger than the corresponding secondary node. As expected, the degree of the primary node tends to grow faster than the corresponding secondary node. The Rare class exhibits the largest difference between primary and secondary nodes, with the average of the first being more than ten times larger (shown in Table 3.11). In contrast, the Co-appearance class shows a smaller difference between primary and secondary, indicating that both nodes grow their degrees over time. Thus, degree evolution is also a distinctive feature both among classes and among node types (primary or secondary), consistently across datasets. Finally, note that the metric has a broad range when considering primary nodes (illustrated by the large boxes), indicating that individuals have quite different behaviors.



(a) (b)

(c)

Figure 3.8: Degree evolution for the nodes (names) associated with the individuals in Figure 3.1. These examples are representative of the different causes of ambiguity, Rare, Swap and Co-appearance.



(a) DBLP (b) Google Scholar

Figure 3.9: Boxplot (25th, 50th, 75th percentile defined in the box) of the derivative of the degree evolution for primary and secondary nodes (names) for each class in each dataset.

3.3 Discussion and conclusion

Despite the various reasons for individuals to appear with multiple names in bibliographic data, we show that a few very distinctive patterns are the cause for most appearances. In particular, assuming knowledge of the multiple names used by individuals, the proposed method constructs time series based on the usage patterns of such names and classifies individuals into one of three classes: Rare, Swap, Co-occurrence. When applied to two real datasets (obtained from DBLP and Google Scholar) our method reveals that these three classes are consistent and representative of the patterns for synonyms use. The Rare class represents the cases where individuals use one name most of the time with the second name appearing in just a few publications. The Swap represents individuals that start using one name but then stop and change to another name. The co-appearance class represents individuals that use two names (or more) regularly and interchangeably. Despite intuitive, these classes indeed consistent and representative of multiple name usage patterns present in real data, with the class Rare being the most common (43% and 53% of individuals in DBLP and Google Scholar, respectively).

Interestingly, we also show that different causes of ambiguity produces different network structures, such as the number of common neighbors (between the primary and secondary names) and the degree evolution of the nodes (names). This finding is also robust across the two datasets and seems inherent to name usage patterns of the individuals. Thus, the fingerprint in the collaboration network produced by ambiguous names is class dependent and very different. The observation name ambiguity across different individuals appears to have different network structure was also made recently, albeit without an explanation [3], while another recent work investigates the impact on the network of different kinds of ambiguity (synonyms and homonyms) [41, 44]. Exploring the specific structural fingerprints of different causes for synonyms might lead to the design of more effective name disambiguation algorithms.

Our approach and method is not constrained to bibliographic data and could be applied to other contexts where name ambiguity is also present. For example, the appearance of different names for the same character in novels or movies may also exhibit distinctive usage patterns.

Chapter 4

A model and a naive algorithm

In this chapter we first propose a probabilistic model to introduce ambiguity in a network by duplicating vertices and adding and removing edges. Then, we propose a simple label-free algorithm to remove ambiguities by identifying duplicate vertices based only in structural features. We evaluate the performance of this algorithm under two classical random network models. Results indicate that such network structure can indeed be used to identify ambiguities, yielding very high precision when local structure is preserved.

Towards this direction, we make the following contributions:

1. Ambiguity model: based on intuition and empirical observations of real data, we propose a probabilistic model that introduces ambiguity in a social network. The model has three intuitive parameters used for tuning the desired amount and structure of ambiguity and can operate over any original social network. This model is presented in Section 4.1.

2. Disambiguation algorithm: again, based on intuition and empirical observations on real data, we propose a simple and efficient label-free algorithm for removing ambiguity. Our algorithm uses only the structure of the network of observed labels but not the labels themselves to identify nodes (labels) that refer to the same person. We present an extensive analysis of the performance (precision and recall) of this algorithm when applying the proposed ambiguity model to random graph models. The algorithm and its evaluation are presented in Sections 4.2 and 4.3, respectively.

4.1 Ambiguation model

In this section we present a novel probabilistic model that introduces ambiguity in a network. The model is mostly tailored for social networks and is based on intuition and empirical observations. The idea is to duplicate nodes and add/remove edges

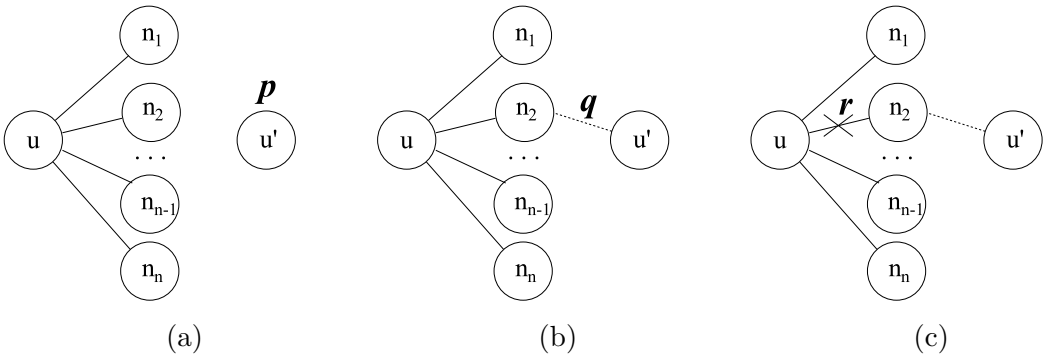


Figure 4.1: Parameters of the probabilistic model for create ambiguity in a network. In (a) vertex duplication phase, (b) edge addition phase, and (c) edge removal phase.

to neighbours of an original node. A duplicated node represents a second label for the original node. Therefore, one object (node) of the original network can be represented by two nodes (labels) in the ambiguous network and relationships among the original object (node) can be copied to its duplicate and removed from itself.

Consider a network represented as a graph $G = (V, E)$ in which V is the set of vertices (e.g., people) and E is the set of edges (e.g., friendship relationship). In this graph, each vertex uniquely identifies an object in the network. The proposed model has three phases, each with a parameter:

1. **Vertex duplication:** with a probability p a vertex is duplicated;

2. **Edge addition:** with a probability q an edge between a neighbour of the original vertex and the duplicated vertex is created;

3. **Edge removal:** with a probability r an original edge that was copied to a duplicated vertex is removed.

In the vertex duplication phase, the vertices are duplicated creating ambiguity. Each vertex $u \in V$ is sampled with probability p independently to generate another graph with a duplicate vertex, u', as shown in Figure 4.1a. Note that p controls the amount of ambiguity introduced in the network, so that with $p = 1$ all vertices will have a duplicate in a network.

In the edge duplication phase, the neighbours from the original vertex are copied to the duplicated vertex. For each neighbour $v \in N_u$(neighbours of u) of an original vertex u that has been duplicated, with probability q independently, an edge $e = (u', v)$ is created as illustrated in Figure 4.1b. Note that with $q = 1$ all neighbours from u will become neighbours of u'.

In the edge removal phase, edges between an original vertex u and a neighbour v, which has become a neighbour of u', is removed with probability r, independently, as shown in Figure 4.1c. Note that for $r = 1$ all edges between the original vertex u and

its neighbours that became neighbours of the duplicate vertex u' will be removed. Algorithm 1 describes this ambiguity model.

Algorithm 1: Model to introduce ambiguity with parameters: p, q, r.

```
Data: G = (V, E), p, q, r
Result: G' = (V', E')
E' ⟵ E ; V_d ⟵ ∅ ; E_d ⟵ ∅
for v in V do
    with probability p, duplicate v into v' and V_d ⟵ V_d ∪ v'
for v' in V_d do
    v ⟵ original(v')
    N ⟵ neighbours(v)
    for u in N do
        with probability q, create e' = (v', u) and E_d ⟵ E_d ∪ e'
        if e' in E_d then
            with probability r, remove e = (v, u) from E'
V' ⟵ V ∪ V_d ; E' ⟵ E' ∪ E_d
```

4.2 Algorithm for removing ambiguities

In this section we present a simple algorithm to identify ambiguities in a social network. In particular, we consider just the case where a single object, due to ambiguities, can be represented in the observed label network by more than one vertex. Our algorithm identifies network nodes that represent the same entity without resorting to label information, i.e., only structure information is used.

We developed several structure-based heuristics to identify nodes in the label network that might represent the same entity. For example, we consider that two nodes might refer to the same entity if they are at distance 2, since it is unlikely that a node will have a relationship with itself using two different labels. Moreover, the same is considered if the common neighbourhood between two vertices strongly overlaps and is contained in one another. We aim at developing a conservative approach to merge nodes, in order to minimize false-positives, allowing greater applicability of the algorithm. The proposed algorithm is described in Algorithm 2.

4.3 Evaluation

In this section we present an extensive evaluation of the performance of the proposed algorithm to remove ambiguities when applied to networks generated by the ambiguity model. The evaluation has the following steps: (i) generate the networks, (ii) introduce ambiguity using the model , (iii) apply the algorithm proposed in Section 4.2 to remove ambiguity and (iv) measure the precision and recall of the algorithm.

Algorithm 2: Algorithm - Remove ambiguity

```
Data: G = (V, E), α
for v in V do
    P ⟵ ∅ ; N_v ⟵ neighbours(v) ;  D_v^2 ⟵ {u|distance(u, v) = 2}
    for u in D_v^2 do
        if degree(v) ≥ α and degree(v) ≤ degree(u) then
            N_u ⟵ neighbours(u)
            if N_v ⊆ N_u then
                P ⟵ P ∪ u
    if sizeOf(P) = 1 then /* Ambiguity found!  Unify v and P.first()
     */
        merge(v,P.first())
```

In order to generate the networks, we use two models, the Erdos-Renyi model, that generates graphs connecting nodes randomly, and the Watt-Strograts model, that generates graphs with small-world properties [45]. Both networks were generated with $n = 100,000$ vertices and average degree of eight (rewiring probability of two percent was used in the Watts-Strogats model).

Next, we introduce ambiguity into the two networks created. We apply the probabilistic model with different values for the parameters p, q and r aiming to evaluate how these parameters affect the identification of duplicated vertices. The values used for each parameter are 0.1, 0.3, 0.5, 0.7 and 0.9. We apply the algorithm to remove the duplicated vertices, with parameter $\alpha = 0$, and we evaluate the performance by measuring the precision and recall of the algorithm. For each parameter configuration, we perform thirty independent runs and report the sample average of performance metrics. The algorithm performance for the Erdos-Renyi and Watts-Strogatz networks models with ambiguity are shown in Figures 4.2 and 4.3, respectively, for all combinations of model parameters.

The precision and recall for the Erdos-Renyi model are shown in Figures 4.2a and 4.2b, respectively. Note that the parameter p is not critical to the algorithm, when ten or ninety percent of the vertices are duplicated the performance of the algorithm remains roughly the same. This occurs because in the Erdos-Renyi network model lacks local structure and, therefore, any duplication of vertices and edges creates a local structure that is detected by the algorithm. In these figures the lines are grouped by the parameter r, so that with smaller values of r we get around 100% of precision and 50% of recall.

In Figures 4.2c and 4.2d we observe an inflection point with respect to parameter q, with precision and recall growing and then decreasing. This occurs because the number of edges that are removed from the original grows with q. However, for lower values of q the duplicated vertex has a small degree and thus there are many

vertices that are candidates to be its original version and the algorithm fails to make a decision yielding a lower precision and recall. The inflection point changes with the value of r because the expected number of removed edges is $d_u qr$ where d_u is the degree of the node u.

Figures 4.2e and 4.2f shows the precision and recall as a function of parameter r, respectively. Clearly, r is the most sensitive parameter for the performance of the algorithm. Note that precision is more than 90% for values of r lower than 0.5, independent of the other parameters p and q. As r grows the precision and the recall decrease as more original edges are removed and the algorithm fails to find the original vertex that corresponds to the duplicated one.

Results under the Watts-Strogatz network model are shown in Figure 4.3. In general, results have the same qualitative trends as for the Erdos-Renyi model, with a higher sensitivity for the parameter r. For example Figures 4.3e and 4.3f illustrate that performance degrades quickly as r increses. This occurs due to the local structure present in the Watts-Strogatz model, which makes the algorithm fail if few edges are removed.

4.4 Conclusion

In this chapter we proposed a probabilistic model that introduces ambiguity in the context of social networks using three parameters for tuning the desired amount of structural ambiguity. We also propose a simple disambiguation algorithm that uses only structure to identify duplicate nodes. We extensively evaluate the performance of the algorithm using random graphs subject to ambiguation introduced by the proposed ambiguity model. Results indicate that the structure of a network can successfully be used to identify ambiguities and does not strongly depend on the amount (fraction) of objects with double identity (duplicated nodes), but on the local structure between the main and the alternative labels. In particular, local network features such as absence of direct edge and common neighbourhood play a key role in disambiguation of social networks.

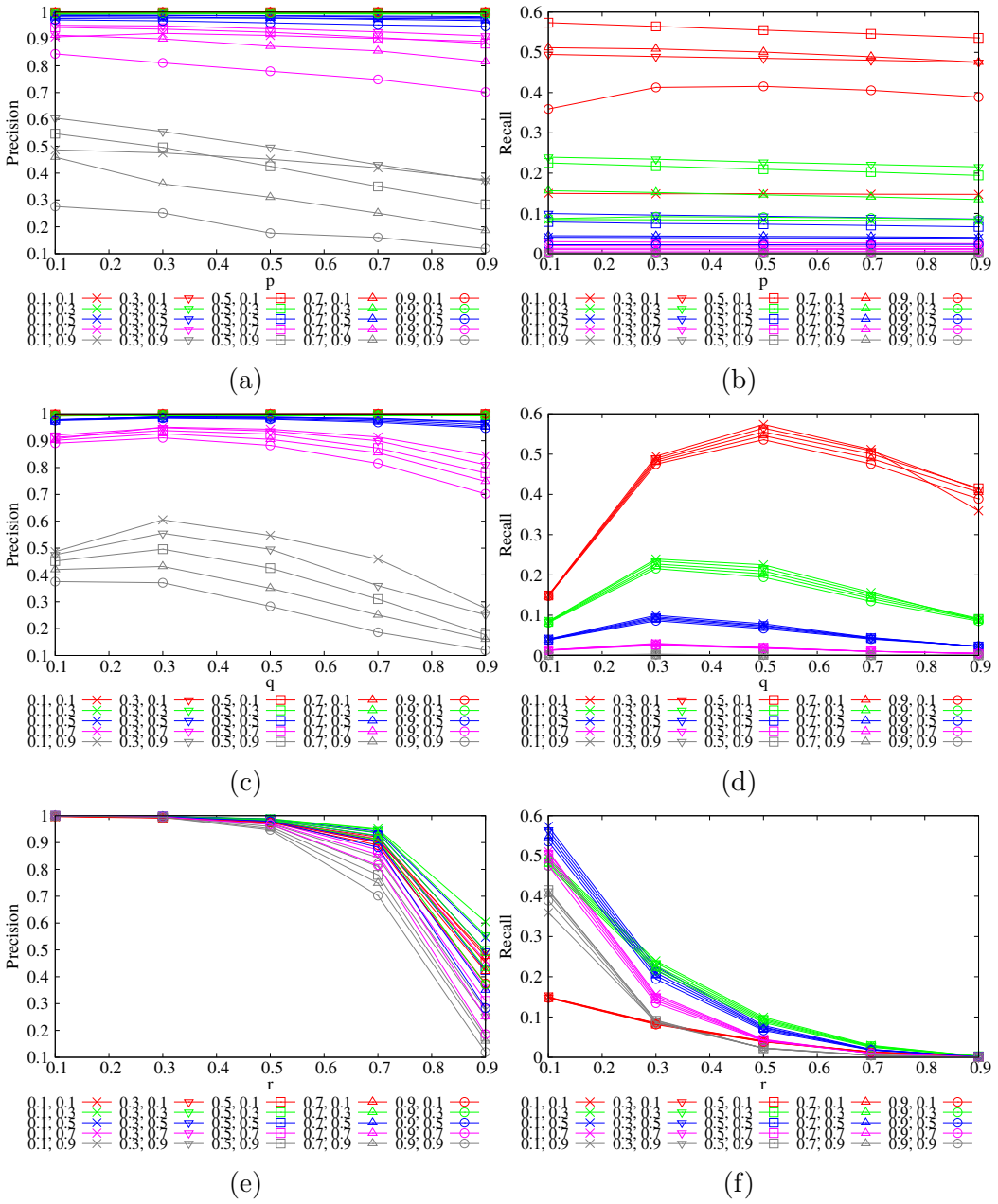


Figure 4.2: Evaluation in Erdos-Renyi network with ambiguity. In (a,c,e) precision and in (b,d,f) recall. The pair of values in the legend correspond to p, q, r with the exception of the value appearing in the x-axis.

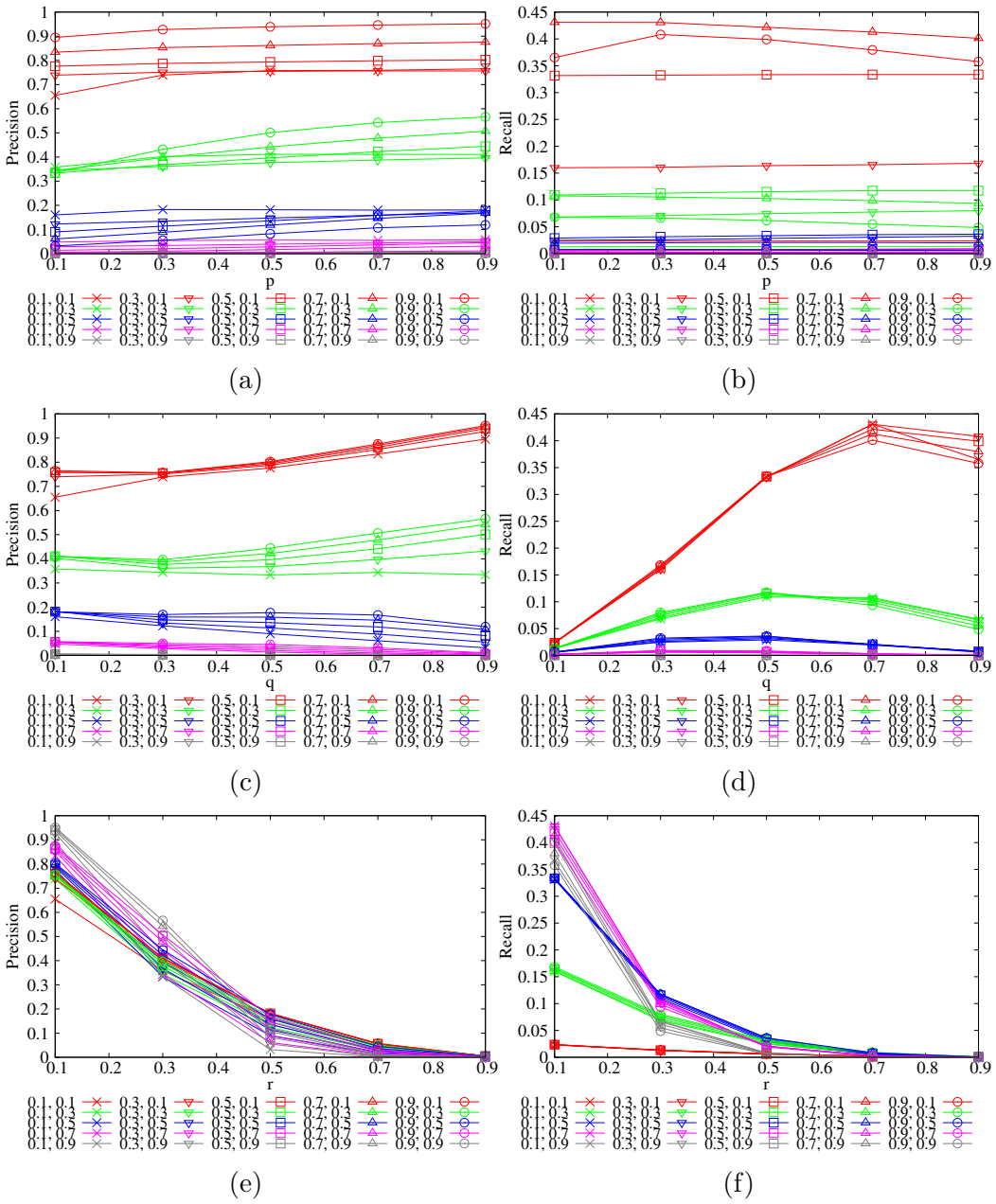


Figure 4.3: Evaluation in Watts-Strogatz network with ambiguity. In (a,c,e) precision and in (b,d,f) recall. The pair of values in the legend correspond to p, q, r with the exception of the value appearing in the x-axis.

Chapter 5

Finding ambiguous nodes in egonets

Many digital libraries have a profile page for authors with some information about them and their publications. For example, DBLP[1], Google Scholar[2] and Lattes[3] all have a specific page for each individual listing all their publications and other information, such as affiliation and projects. Specifically in Google Scholar and Lattes the author must include or verify their publications and give permission to make the profile public. Although these profiles contain publications of an individual. Different names for this author appear in its publications, leading to the synonym name ambiguity problem.

In this chapter we consider the problem of finding the names used by an individual across its publication as listed in its profile on Google Scholar. In particular, we build an anonymous egonet with edge weights and propose an algorithm that considers only the network structure to discover all the ambiguous nodes. The algorithm does not use any label information as seen in the related work described in chapter 2.

We make the following contributions:

1. Build an anonymous egonet with edge weights that encode structural information about the author (profile owner);

2. Design and implement an algorithm based on the network structure to identify the multiple nodes in the egonet that correspond to the individual that owns the profile.

[1]http://dblp.uni-trier.de
[2]https://scholar.google.com
[3]http://lattes.cnpq.br

5.1 Problem statement

Consider an individual and the corresponding profile that lists publications co-authored by the individual. Let $\mathcal{P}$ denote the set of publications in the profile, where $|\mathcal{P}|$ denotes the number of publications. Let $\mathcal{L}_i$ denote the set of labels corresponding to the co-author names of publication $p_i \in \mathcal{P}$. Note that in every publication there is exactly one label that corresponds to the profile owner. Let $\mathcal{L} = \bigcup_{i=1}^{|\mathcal{P}|} \mathcal{L}_i$ denote the set of labels that appear across all publications in the profile with $n = |\mathcal{L}|$ denoting the number of (different) labels. Without loss of generality we enumerate the labels from 1 to n, and thus have $l_1, l_2, \ldots, l_n$.

The problem that we address is to determine the set of labels that correspond to the profile owner. Moreover, we assume the labels themselves (i.e., the string corresponding to the name) have no information about the profile owner or the co-authors, and can thus be viewed as a random number. Note, however, that the sets $\mathcal{L}_i$ provide the only information that can be exploited to craft a solution. But the amount of information available will depend on the sets. For example, if every label in $\mathcal{L}$ appears exactly once across all publications, then there is no information to be exploited. At the same time, if a single label l_1 appears in all publications (i.e., every set $\mathcal{L}_i$) while all other labels appear just once, then very likely l_1 is the only label that corresponds to the profile owner. In practice, the sets $\mathcal{L}_i$ will be diverse across profile owners.

Note that, by ignoring the content of the names (i.e., the name string), we focus solely on the co-authorship *structure*. In a sense, our goal is to determine the power of the structure in resolving name ambiguities in social networks. Can *structure* alone help? As we will soon show, the answer to this question is quite positive.

5.2 The method

`EgoIds` starts by constructing an *egonet* for every profile owner. In particular, every profile will be treated separately and independently from one another, so we focus the discussion on a single profile. Using the publication information in the profile, the *egonet* is constructed as follows. Let $G = (V, E)$ be an undirected weighted graph with vertex set $V = \{1, 2, \ldots, n\}$ where vertex $i \in V$ corresponds to label l_i in $\mathcal{L}$. Thus, note that every label has a corresponding node in the graph. Now every set $\mathcal{L}_i$ created a clique in G with the nodes corresponding to the labels that appear in $\mathcal{L}_i$. Thus, G is a superposition (i.e., union) of cliques, one for each publication p_i. Thus, $(x, y) \in E$ if and only if $l_x, l_y \in L_i$ for some publication $p_i \in \mathcal{P}$.

Three different weights are associated with each edge $e = (i, j) \in E$. These weights will soon be used in the algorithm that identifies the nodes that correspond

to the profile owner. The weights are defined as follows:

$$w1_{i,j} = \sum_{k=1}^{|\mathcal{P}|} \mathbb{1}(l_i, l_j \in \mathcal{L}_k) \tag{5.1}$$

$$w2_{i,j} = \sum_{k=1}^{|\mathcal{P}|} \frac{\mathbb{1}(l_i, l_j \in \mathcal{L}_k)}{|\mathcal{L}_k| - 1} \tag{5.2}$$

$$w3_{i,j} = \sum_{k=1}^{|\mathcal{P}|} \frac{\mathbb{1}(l_i, l_j \in \mathcal{L}_k)}{\binom{|\mathcal{L}_k|}{2}} \tag{5.3}$$

where $\mathbb{1}(\cdot)$ is the indicator function. Intuitively, $w1_{i,j}$ denotes the number of publications where labels l_i and l_j appear as co-authors, whereas $w2_{i,j}$ and $w3_{i,j}$ denote the normalized collaboration strength between labels l_i and l_j with the former penalizing more large sets of co-authors. Moreover, $w2$ and $w3$ can be used to recover other aspects concerning the labels and publications. For example, for any $i \in V$, $\sum_j w2_{i,j}$ corresponds to the number of publications where l_i is listed as a co-author. Also, $\sum_{i,j} w3_{i,j}$ corresponds to the number of publications in the profile, namely $|\mathcal{P}|$. These observations will be used soon.

To exemplify lets build the *egonet* of John Von Neumann using his Google Scholar profile page[4]. Table 5.1 shows the publications that will be considered in this example. For each author name appearing in the publications, an identifier is given to him (shown in Table 5.2a). Note that three different identifiers are given to John Von Neumann since he appears with three different labels. Next, the edge weights are calculated (shown in Table 5.2b). To illustrate, consider the edges created by the first publication of Table 5.1: (1,3), (1,2) and (2,3). The edge weights corresponding to this publication are given by $w1_{1,3} = w1_{1,2} = w1_{2,3}= 1$, $w2_{1,3} = w2_{1,2} = w2_{2,3} = 0.5$ and $w3_{1,3} = w3_{1,2} = w3_{2,3}= 0.33$. After all publication in table are considered, the weights of edge (1,3) will correspond to $w1_{1,3}$=2, $w2_{1,3}$=1.5, $w3_{1,3}$=1.33. The John Von Neumanns egonet is shown in Figure 5.1. Note that nodes 3, 7 and 11 represent John Von Neumann while all other nodes correspond to collaborators. Moreover, the *egonet* has two connected components.

5.2.1 The algorithm

Recall that the profile owner is a co-author in every publication in the profile and thus appears exactly once in each publication. How can this information be exploited to identify labels of the profile owner?

Consider an *egonet* and the following statement: "Every node that represents a profile owner is at distance two from at least another node that also represents

[4]https://scholar.google.com.br/citations?user=6kEXBa0AAAAJ

Table 5.1: Some of John Von Neumann's publications listed in his Google Scholar profile.

Authors	Title
AW Burks, HH Goldstine, **J Von Neumann**	Preliminary discussion of the logical design of an ...
S Chandrasekhar, **J Von Neumann**	The Statistics of the Gravitational Field Arising from a Random ...
BI Hart, **John von Neumann**	Tabulation of the probabilities for the ratio of the mean ...
BO Koopman, **J V Neumann**	Dynamical systems of continuous spectra
HH Goldstine, **J V Neumann**	Blast wave calculation
J Von Neumann, RH Kent, HR Bellinson, BI Hart	The mean square successive difference
D Hilbert, **J Neumann**, L Nordheim	Uber die grundlagen der quantenmechanik
R Zeller, **J Neumann**	Calibration-test member for a coordinate-measuring instrument
J Von Neumann, AW Burks	Theory of self-reproducing automata

Table 5.2: In (a) the identifiers of the co-authors from John Von Neumanns publications and in (b) the corresponding edge weights.

(a)

$Name$	id
AW Burks	1
HH Goldstine	2
J Von Neumann	3
S Chandrasekhar	4
BI Hart	5
BO Koopman	6
J V Neumann	7
RH Kent	8
HR Bellinson	9
D Hilbert	10
J Neumann	11
L Nordheim	12
R Zeller	13

(b)

$Edge(id1, id2)$	$Weights[w1, w2, w3]$
(1,2), (2,3)	[1,0.5,0.33]
(1,3)	[2,1.5,1.33]
(4,3), (5,3), (6,7), (2,7)	[1,1.0,1.0]
(3,8), (3,9), (3,5), (8,9), (8,5), (9,5)	[1,0.33,0.17]
(10,11), (10,12), (11,12)	[1,0.5,0.33]
(13,11)	[1,1.0,1.0]

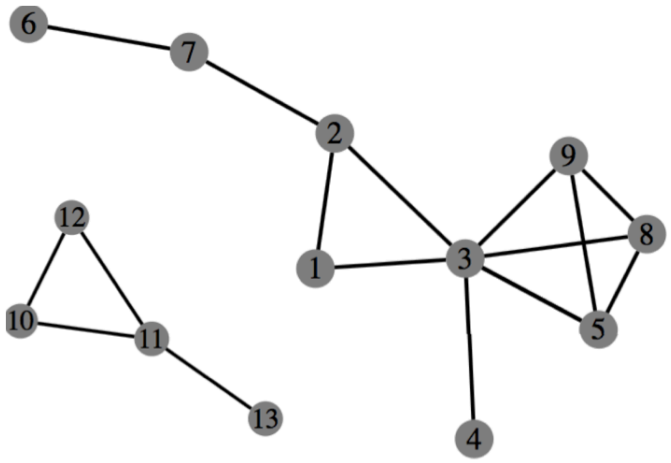


Figure 5.1: Example of part of John Von Neumann's *egonet* as defined by Google Scholar. Note that nodes 3, 7 and 11 correspond to John Von Neumann while all other nodes correspond to collaborators.

the profile owner". Under our assumptions, this statement is true. Consider a node i that corresponds to the profile owner. Consider the connected component of node i and assume it has at least one other node at distance 2. These nodes must have appeared in a publication with at least one common neighbor of i (necessary condition to be as distance 2). Since the profile owner is in every publication, one of these nodes must corresponds to the profile owner. Thus, we can use this observation to determine the set of nodes that could correspond to the profile owner.

Let $\mathcal{S}$ denote a set of nodes that could correspond to the profile owner of an *egonet* defined by $G = (V, E)$. Then the following must be true:

1. For every $i \in V$, either $i \in \mathcal{S}$ or there exists an edge $(i, j) \in E$ such that $j \in \mathcal{S}$.
2. For every $i, j \in \mathcal{S}$, edge $(i, j) \notin E$.
3. If $|\mathcal{S}| > 1$, then for every $i\mathcal{S}$, there exists $j \in \mathcal{S}$ such that $d(i, j) = 2$, where $d(i, j)$ is the hop distance between nodes i and j.

Note that statement 1 is identical to the definition of a dominating set in a graph. Statement 2 is identical to the definition of an independent set in a graph. Thus, $\mathcal{S}$ is an independent and dominating set of G. However, statement 3 adds an additional constraint to nodes in the independent and dominating set. Computing the minimal dominating set that is also an independent set of an arbitrary graph G is known to be NP-Complete problem [46]. Moreover, there could be various solutions. In fact, we would like to find the set $\mathcal{S}$ that *most likely* corresponds to the exact set of labels of the profile owner. Note that the minimal independent and dominating set, even if unique, may not be the correct solution!

In order to tackle this problem, we propose the following greedy algorithm to construct the set $\mathcal{S}$. Using some domain knowledge about the problem (encoded in

the weights and structure) we hope to generate a set $\mathcal{S}$ that is likely to have the nodes corresponding to the profile owner. The algorithm works as follows. Let S_c denote the set of nodes that can still be chosen to compose the set $\mathcal{S}$. Note that S_c is the current set of candidate nodes for $\mathcal{S}$ and initially, let $S_c = V$, and the set of nodes corresponding to the owner $\mathcal{S} = \varnothing$.

1. Choose the node with highest degree in S_c and include it in the set $\mathcal{S}$, remove all its neighbors from the set S_c;

2. Choose a node from S_c that is at distance two from at least one of the nodes in $\mathcal{S}$. If there is more than one option, use the following list of preferences:

   (a) Choose the nodes with the largest degree;

   (b) From that set, select the node with more common neighbors with nodes already in $\mathcal{S}$;

   (c) From that set, select the node that the edges have a higher sum of weight $w2$ (across all edges incident to the node);

   (d) From that set, select the node with lowest sum of weight $w3$ (across all edges incident to the node).

   Note that largest weight $w2$ means more publications (sum of $w2$ is equal to number of publications with at least one co-author) and smallest sum $w3$ means to give priority to the authors that collaborate with more authors.

3. If it is not possible to distinguish between the selected vertices, stop the algorithm returning its partial set of owners $\mathcal{S}$. For example, when nodes i and j appear together in all publications of an individual they are structurally identical and it is impossible to distinguish between them.

4. Repeat steps 2 and 3 until S_c is empty, and return the nodes in $\mathcal{S}$.

Note that different connected components will be treated identically and independently. Thus, the above algorithm is really applied to each connected component of the *egonet*. A more precise pseudo-code of the algorithm is given below.

When the algorithm is applied to the example shown in Figure 5.1, it first selects node 3 (largest degree), then node 7 (only node at distance 2 from node 3), and then node 11 (largest degree in the second connected component). Note that these three nodes correspond exactly to the profile owner, and thus the algorithm has perfect precision and perfect recall in this example.

The worst case time complexity of the algorithm is $O(nd^{*2})$, where d^* is the largest degree in the graph. Note that the set D_2 has size at most d^{*2} and the inner loops traverse this set once. The outer loop (including nodes in the set $\mathcal{S}$) runs for

Algorithm 3: EgoIds

```
Data: Egonet G = {V, E} and corresponding edge weights
S ← ∅;
foreach Connected component c, c ∈ G do
    S_c ← all nodes in c;
    u ← node with the highest degree in S_c;
    add u into S;
    remove u and all its neighbors from S_c;
    while S_c ≠ ∅ do
        D_2 ← {u ∈ S_c | ∃v, v ∈ S, distance(u, v) = 2};
        if |D_2| = 0 then
            continue to next connected component;
        else if |D_2| > 1 then
            remove from D_2 all nodes with degree < Δ(D_2);
            if |D_2| > 1 then
                foreach u_i ∈ D_2 do
                    n_i(u_i) ← sum of common neighbours between u_i and each
                     node in S;
                remove from D_2 all nodes u_i such that n_i(u_i) < max(n_i);
                if |D_2| > 1 then
                    foreach u_i ∈ D_2 do
                        w2_i(u_i) ← Σ_{u_i∈e} w2(e);
                    remove from D_2 all nodes u_i such that w2_i(u_i) < max(w2_i);
                    if |D_2| > 1 then
                        foreach u_i ∈ D_2 do
                            w3_i(u_i) ← Σ_{u_i∈e} w3(e);
                        remove from D_2 all nodes u_i such that
                         w3_i(u_i) < max(w3_i);
                        if |D_2| > 1 then
                            continue to next connected component;
        u ← only element from D_2;
        add u into S;
        remove u and all its neighbours from S_c;
return S;
```

at most n steps. In practice, the algorithm runs very fast because n (size of *egonets*) is rather small as we will see in the following section.

5.3 Empirical evaluation

In this section, we present experimental results that demonstrate the effectiveness, efficiency and practicability of our method. First we describe in details the datasets used and then we show the results obtained when applying `EgoIds` to discover the identities of the profile owners in the datasets.

5.3.1 Datasets

We used two different real bibliographic records that are used to build collaboration networks and are readily available online: the DBLP[5] and a small subset of Google Scholar[6] that we collected.

DBLP

In particular, we considered three different collections from this database.

The first is the entire DBLP[7] version of May 2014. This version of DBLP listed 14,290 individuals with more than one name, and a total of 29,126 different names among them.

A collection derived from DBLP with 477 individuals collected and used by Santana et al. [26]. The individuals in this dataset were manually labelled based on the individuals's publication home page, affiliation, e-mail, and co-author names in a complete name format. This collection will be called DBLP-UFMG.

The last dataset is a collection derived from DBLP referred by KISTI[8] built by the Korean Institute of Science and Tecnology Information for English homonym author name disambiguation, [47]. The top 1,000 most frequent author names from a late-2007 DBLP version were obtained jointly with their bibliographic records (i.e., publications). This collection has 6,908 individuals and 41,659 name instances.

Google Scholar

In order to select profiles in Google Scholar, we considered all individuals that were awarded a research fellowships from CNPq[9]. We chose this target group since

[5]http://dblp.uni-trier.de

[6]http://scholar.google.com

[7]Link to download the XML of the entire DBLP database: http://dblp.uni-trier.de/xml/

[8]http://www.kisti.re.kr

[9]CNPq is the Brazilian National Research Council responsible for funding research, similar to the National Science Foundation (NSF) in the United States of America.

Brazilians tend to have many first and last names, and use them rather freely in publications.

From the 13,473 researchers that receive a research fellowship from CNPq, 7624 have a public Google Scholar profile (we automatically search for their public profiles on Google Scholar using their names). We collected all publications available in the profile for each individual and found 7089 profiles where the listed publications exhibit more than one name for the profile owner.

5.3.2 Dataset characterization

In what follows we first characterize the different datasets in order to illustrate their differences. In particular, we consider features such as the number of publications per individual and the number of names per individual. We also looked at the structural features of the *egonets* across the different datasets, such as their size and number of connected component.

Consider the number of publications per author in each dataset. Figure 5.2 show sthat the datasets DBLP-UFMG and KIST, both used in [26], have a very small number of publication per author. Around 80% of the individuals in the KISTI dataset and 60% in the DBLP-UFMG dataset have less than ten publications, while for Google Scholar dataset around 80% have more than 100 publications. As we can see there is a big difference in the number of publications per profile in the datasets. Since `EgoIds` uses only the relationships among collaborations to reveal the various identities of the profile owner, having more publications is better since it will better reflect the actual *egonet* of the profile owner. When there are few publications, the *egonet* has little information and thus the algorithm may not correctly identify nodes corresponding to the profile owner.

Table 5.3 shows the number of individuals with exactly k names in each of the datasets. Note that the datasets used by Santana et al. [26] has fewer names per author in contrast with the DBLP-GT and the Google Scholar datasets. In the Google Scholar dataset, most of the individuals have 3 or more names meaning that there are more ambiguity in this dataset and, thus, posing a harder problem to solve. In the DBLP-GT dataset all individuals have more than one name because the dataset is constructed only with individuals that the DBLP has recognized as having duplicated names. In both DBLP-UFMG and KISTI datasets the majority of the individuals have a single name. This is because Santana et al. [26] focus on the homonym ambiguity problem where the goal is to find names that represent different individuals.

Next we considered the network structural features to characterize the datasets. First, we look at the size of the *egonets* as measured by the number of nodes.

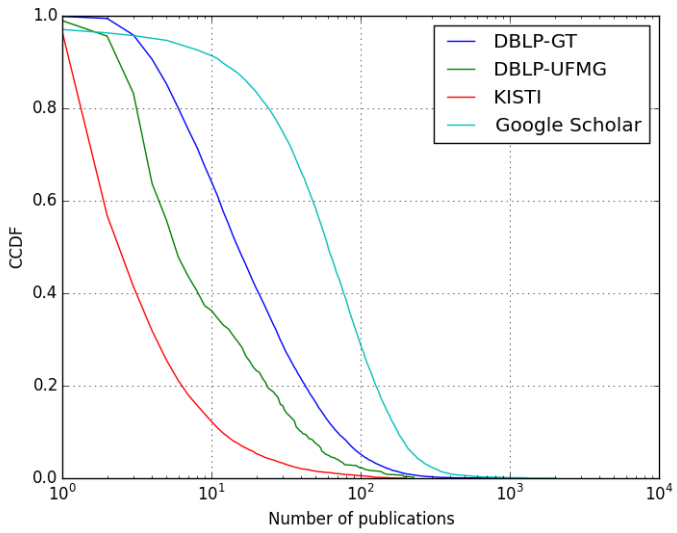


Figure 5.2: CCDF (Complementary Cumulative Distribution Function) of the number of publications in a profile for different datasets.

Table 5.3: Number of profile owners with k names.

k	DBLP-GT	DBLP-UFMG	KISTI	Google Scholar
1	0	267	6296	1249
2	13700	163	544	1853
3	569	35	55	1422
4	21	8	13	950
5 or more	0	4	0	2150

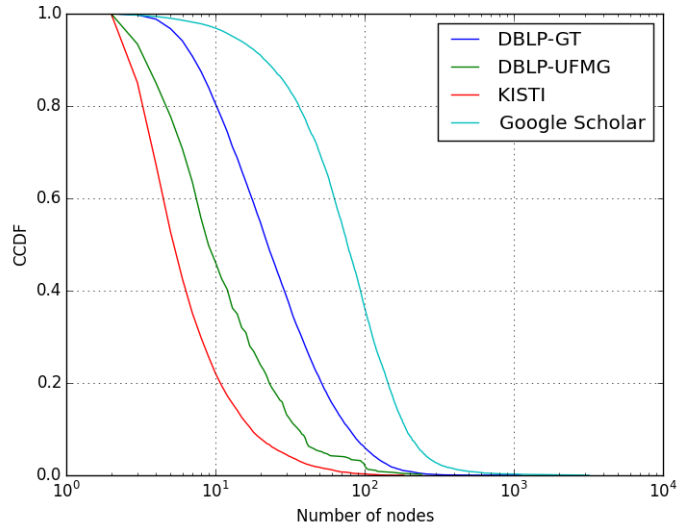


Figure 5.3: CCDF of the size of egonets (number of nodes).

Table 5.4: Number of individuals with egonets that have exactly k connected components .

k	DBLP-GT	DBLP-UFMG	KISTI	Google Scholar
1	12623	448	6358	4344
2	1624	23	289	1823
3	24	1	19	620
4	0	0	3	245
5 or more	0	0	0	366

The number of nodes represents the number of labels among the co-authors of the profile owner (including the multiple labels of the profile owner). Figure 5.3 shows the CCDF of the size of the *egonets* for all datasets. The KISTI dataset has the smallest *egonets* where 80% of them has less than 10 nodes in contrast with the *egonets* from the Google Scholar dataset where 35% of the individuals have more than 100 nodes. Table 5.5 shows the average and the standard deviation of the size of the *egonets* and we can see that the smallest *egonets* are from DBLP-UFMG and KIST datasets.

Table 5.4 shows the number of individuals with *egonets* that have exactly k connected components. For all datasets, the majority of the individuals have one *egonet* with a single connected component. With the exception of Google Scholar, a relative small fraction of individuals have *egonets* with more than one connected component. In Google Scholar, since many more publications were collected for each individual, we expect that the larger number of labels per individual will also generate a larger number of connected components.

Table 5.5: Size of the egonets (number of nodes) and fraction of nodes that belong to the largest connected component (LCC).

Database	Size of egonet	Fraction of nodes in LCC
DBLP-GT	34.72 ± 39.44	0.97 ± 0.09
DBLP-UFMG	16.55 ± 23.68	0.99 ± 0.06
KISTI	8.41 ± 11.75	0.99 ± 0.07
Google Scholar	100.89 ± 129.31	0.94 ± 0.11

The last characterization considers the fraction of nodes that belong to the largest connected component. Although the *egonet* of many individuals have more than one connected component, the size of these components are not similar. In fact, the majority of the *egonets* have one very large connected component, as show in Table 5.5. The table shows the average and standard deviation across the individuals. Note that for all datasets, more than 90% of the nodes of the *egonet* are in the largest connected component. Again, Google Scholar has some individuals with fewer nodes on its largest connected component, leading to a smaller average.

5.3.3 Evaluation

The `EgoIds` algorithm outputs a set of vertices that represent the nodes of a *egonet* that correspond to the profile owner. Of course, this set may not be correct in the sense that vertices identified by `EgoIds` may not correspond to the profile owner (precision) as well as nodes that do correspond to the profile owner are not identified by the algorithm (accuracy). In this section we evaluate the performance of `EgoIds` on the datasets considering precision, recall and F1 measure. We also compare `EgoIds` results on DBLP-UFMG and KIST datasets with the results of Santana et al. [26].

Note that an egonet may be symmetric in the sense that all nodes are identical with respect to the structural features used by the EgoIds. For example, if two or more nodes have the same largest degree and no other structural differences with respect to edge weights, then the algorithm does have a clear starting point. In this cases, EgoIds does not start and returns an empty set of nodes (as opposed to simply guessing). This situation clearly occurs if a profile has a single publication or if all publications have the same set of co-authors.

Moreover, structural symmetries may also occur during the execution of EgoIds, after a set of nodes have been already identified by the algorithm. In this case, the algorithm also stops execution and returns a partial cover, as opposed to a full cover. Again, faced with structural symmetries the algorithm stops instead of randomly guessing. As we will see, both situations (not starting and partial cover) occur in

Table 5.6: About the algorithm and the datasets: total number of individuals, number of individuals that the algorithm does not start, number of individuals that the algorithm produces a partial cover and a full cover.

	DBLP-GT	DBLP-UFMG	KISTI	Google Scholar
Number of individuals	14,290	477	6,908	7,624
Algorithm does not start	1,416(9.90%)	163(34.17%)	4,664(67.52%)	535(7.01%)
Individuals without co-authors	19(1.34%)	5(3.07%)	239(5.12%)	226(42.24%)
Individuals with only one publication	63(4.45%)	16(9.82%)	2,744(58.83%)	56(10.47%)
Algorithm produces partial cover	2,762(21.45%)	77(24.76%)	135(6.02%)	1,860(26.24%)
Algorithm produces full cover	10,112(78.55%)	237(75.24%)	2,109(93.98%)	5229(73.76%)

real datasets.

Table 5.6 shows for each dataset the total number of individuals, the number of individuals for which the algorithm does not start and the number of individuals for which the algorithm produces a partial cover and a full cover. Considering the number of individuals, the DBLP-GT is the largest dataset with 14,290 individuals while the DBLP-UFMG has only 477. Google Scholar and KIST datasets have around 7,000 individuals. For some individuals the algorithm does not start due to structural symmetries, and thus produces no output (these cases are not considered in the evaluation for precision and recall). However, this occurs for less than 10% of the individuals for the DBLP-GT and Google Scholar datasets, but this number is quite large for DBLP-UFMG and KIST, being over 50% in the latter. The percentage is high for DBLP-UFMG and KIST datasets because the number of publications per individual is very small, as shown in Figure 5.2 and, consequently, the egonets in these datasets are also smaller, as shown in Figure 5.3. When the algorithm starts it may finish producing a full cover, according to the rules for the set. In fact, a full cover is generated for more than 73% of the individuals in each datasets. In the remainder of the cases, the algorithm produces a partial cover meaning that only a fraction of the egonet is covered, due to structural symmetries.

For each individual in the datasets that the algorithm starts, we calculate the precision, recall and the F1 score (F-measure). The F1 is calculated using the precision and the recall as follows:

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{5.4}$$

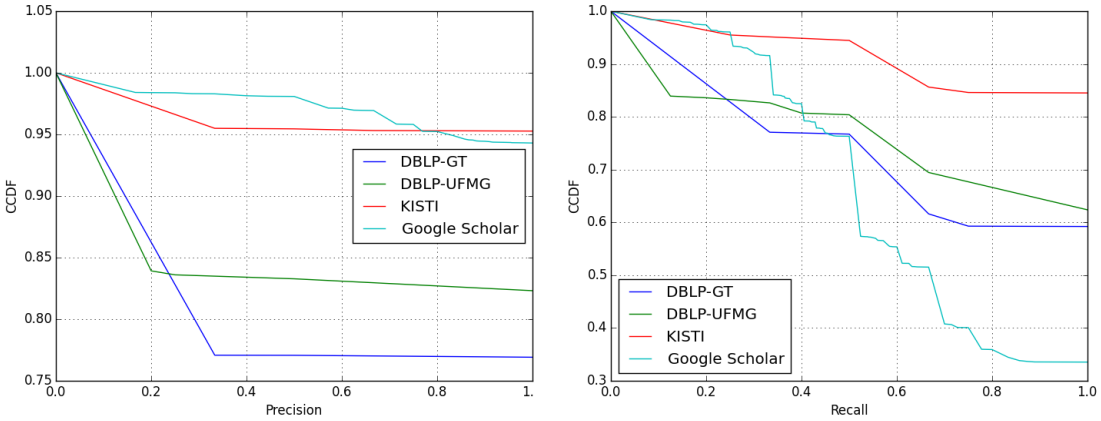
Table 5.7: The average and the standard deviation for the precision, recall and F1 scores of the individuals in all datasets.

Dataset	Precision	Recall	F1	F1 (in [26])
DBLP-GT	0.77 ± 0.42	0.68 ± 0.41	0.71 ± 0.41	n/a
DBLP-UFMG	0.83 ± 0.37	0.74 ± 0.38	0.77 ± 0.37	0.92 ± 0.02
KISTI	0.95 ± 0.21	0.89 ± 0.25	0.92 ± 0.23	0.82 ± 0.01
Google Scholar	0.97 ± 0.15	0.66 ± 0.28	0.76 ± 0.22	n/a

Table 5.7 shows the value for the average and standard deviation for the precision, recall and F1 score across all individuals in all datasets. The precision is more than 77% for all datasets, and for KIST and Google Scholar the results are more than 95% on average meaning that the proportion of the nodes that the EgoIds algorithm suggests to be consolidated as the same identity were in more than 95% of the cases correct. For the recall measures, the proportion of all identities that were correctly identifies by the algorithm. The KISTI dataset is the one with highest recall, 89% on average, and the DBLP-UFMG dataset also showed a high value with around 74%. For the DBLP-GT and Google Scholar datasets the recall is around 66% and that is because these two datasets have the most names per author, as seen in Table 5.3, and thus when this algorithm does not find all the identities then the recall decreases. The F1 measure combines precision and recall and is more than 71% on average for all datasets. For the KIST dataset the value is 92% and performed better than the algorithm proposed in [26]. It is important to notice that EgoIds considers only the network structure and no other information, while in [26] the name string and other informations such as affiliation and email address are used during disambiguation.

Figure 5.4 shows the CCDF of the precision and recall for all datasets. Note that the precision is up to 75% for all individuals in all datasets representing that the in more than 75% of the cases the identities found by the algorithm were correct. The precision for Google Scholar is excellent, more than 94% for all individuals, and for KIST is more than 95%. The recall is good for the KISTI dataset, up to 80%, but it is not that high for DBLP-GT and DBLP-UFMG, which have almost the same behavior with around 20% of the individuals having less than 40% of the recall. Google Scholar present different result from the others, with 70% of the individuals having a recall higher than 50% but for some individuals the recall is less than 35%. This occurs for those individuals with many names (nodes in the egonet). In Google Scholar around 30% of the individuals have five or more names, as shown in Table 5.3.

The CCDF of the F1 score is shown in Figure 5.5. For the KISTI dataset the F1 score is high because its precision and recall are also high. For DBLP-GT and



(a) (b)

Figure 5.4: CCDF of precision (a) and recall (b) for all datasets.

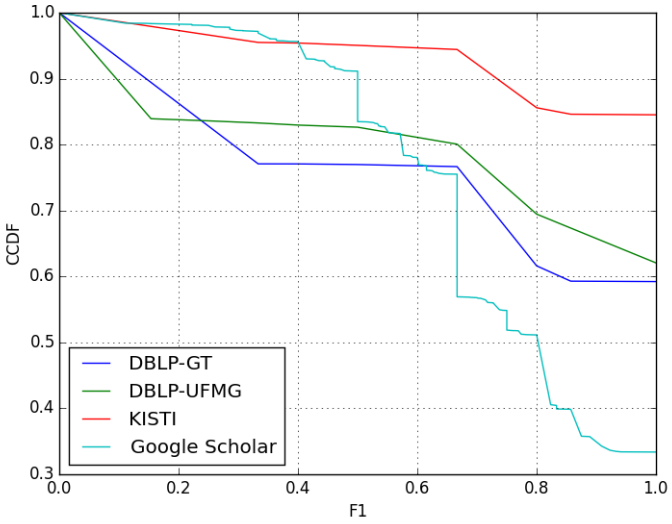


Figure 5.5: CCDF of F1 measure for all datasets.

DBLP-UFMG the F1 scores are quite similar and for around 80% of the individuals in these datasets the algorithm was able to achieve more than 65% of F1 score. Google Scholar has more than 70% of the individuals with more than 65% of F1 but there are some individuals with F1 lower than 40%. This occurs due the recall curve for Google Scholar that drives the trend for F1 measure.

5.4 Conclusion

In this Chapter we proposed an algorithm based on vertex cover to find the ambiguous nodes of an egonet. The algorithm finds a cover by looking only at structural properties of the egonet such as degree, number of neighbours and the edge weights.

To evaluate the algorithm we collected a subset of the Google Scholar dataset and measured the size of the covers and the quality of them.

The results obtained by this work were considered extremely satisfactory and show that the proposed algorithm can be useful in problems related to correctly identifying author's names among bibliographical data. It also presented a high success rate in solving these types of name ambiguity.

Chapter 6

Rooted subgraph matching on two-layer neural network

The continuous increase in privacy concerns by both individuals and enterprises is bound to increase the amount of data that is only available anonymously. In particular, the removal of personally identifiable information (PII) is becoming a common practice by both individuals and enterprises when releasing data into the public realm. In this scenario, classic data science problems that leverage contextual information in their solution must be revisited. A prominent example is entity resolution (aka. name disambiguation) where the goal is to match references (names) to objects (entities) within a large (now anonymized) dataset [2, 33].

A particular but important scenario are social networks which, are often only made available without any PII. Here nodes represent pseudonyms (i.e., random labels) of individuals and edges represent some relationship, such as friendship or collaboration. A fundamental problem when considering large social networks generated from heterogeneous data is name ambiguity. In particular, multiple network nodes may refer to the same individual. This can occur if the individual appears in the original data with multiple names, in which case each name can become a different node in the network. Figure 6.1 illustrates this scenario with a real example of a collaboration network.

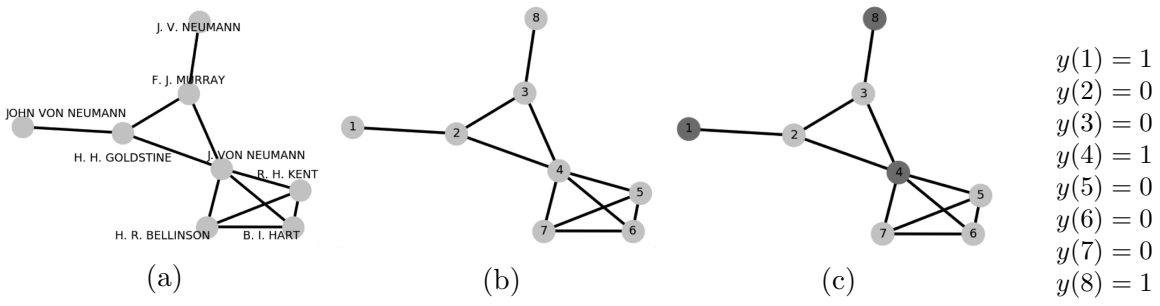


Figure 6.1: Ambiguous nodes identified

Duplicates often occur in unstructured and heterogeneous data. For example, consider bibliographic data from one or more digital libraries, or medical records from one or more hospitals. In such datasets, a single individual can appear with multiple names, for example "R Baeza-Yates" and "RA Baeza-Yates" both refer to the Chilean computer scientist in Google Scholar. When data is anonymized and a social network is released such name instances become different nodes with random labels. Can these duplicate nodes be identified in the anonymous social network?

A significant challenge in tackling this problem is the lack of positive instances (i.e., a set of nodes identified as duplicates). In fact, there is usually no available positive instances when a given social network is released anonymously. Thus, duplicates must be identified using a model that has *never seen* the given social network. In such scenario, inductive learning is the only alternative for tackling duplicate identification. Fortunately, social networks have particular local structures that are related to high level node characteristics and similar across different networks. Such information can be leveraged to learn an effective classification function to identify duplicate nodes (a high level characteristic).

For example, consider a social network where a set of duplicate nodes have been identified, possibly before the network was anonymized, using PII. Local structural features can be extracted from these nodes to serve as a signal for duplicate nodes that depends only on the structure. The same features can be extracted for non-duplicate nodes. These examples can then be used to train a classification model that can be applied to identify duplicate nodes in *any* social network of the same relationship (i.e., collaboration).

The main contribution of this chapter is an inductive learning framework to identify duplicate nodes in anonymized social networks. Our method is indicated by Rooted Subgraph Matching on Two-Layer Neural Network (`RSM2NN`). In particular:

- We propose a fundamental and novel framework for characterizing local structures around nodes (i.e., rooted unlabelled subgraphs with up to 5 nodes). This procedure requires the alignment of local subgraphs with a reference graph (graph isomorphism) to retain consistency. We show how to perform graph alignment efficiently using equivalence classes under automorphism.

- We propose a two-level neural network that learns to identify duplicates from features of the subgraphs. The first level learns to identify duplicates from the point of view of a single subgraph (and for every subgraph). The second level integrates the classification of all subgraphs to learn to identify duplicates.

- We evaluate our framework on two kinds of social networks: collaboration networks and family networks. For collaboration networks we consider 3000 different ego-centered social networks extracted from Google Scholar, as well

as a single social network extracted from DBLP with over 1 million nodes. For family networks we consider an NIH dataset with 162 different family networks and Wikidata with 255. Using nodes from just one network for training, the framework has an median AUC of 97% and 83% when identifying duplicates in networks that it has never seen in the Google Scholar and NIH datasets, respectively.

The remainder of this Chapter is organized as follows. Section 6.1 presents the problem statement. Section 6.2 describes the proposed framework, `RSM2NN` . The four different datasets that are used to evaluate `RSM2NN` are described in section 6.3 together with the different experiments. Section 6.4 presents a brief conclusion.

6.1 Problem statement

The main goal of this work is to identify duplicate nodes in anonymous social networks, a problem we precisely define below. Let $G = (V, E)$ denote an network with node set V and undirected edge set E. Nodes in V are enumerated such that each node corresponds to a unique natural. Thus, $V = \{1, 2, \ldots, n\}$ where $n = |V|$ is the number of nodes in the network.

Since G is a social network, every node maps to an individual. Let $x_G : V \to \mathbb{N}$ denote the identity (e.g., social security number) of the person that corresponds to node $u \in V$. We assume that identities are unique such that $x_G(u) = x_G(v)$ if and only if nodes u and v correspond to the same individual. Let $y_G : V \to \{0, 1\}$ be an indicator function that determines if node $u \in V$ has a duplicate. In particular,

$$y_G(u) = \begin{cases} 1 \text{ , if } \exists\, v \neq u : x_G(u) = x_G(v) \\ 0 \text{ , otherwise} \end{cases} \tag{6.1}$$

Since network G is anonymous we assume there is information between the node label v and its identity $x(v)$. We also assume there is no other side information about G, such as edge weights or edge labels or node classes. The only available information is its edge set E.

Consider a set of social networks $G_1, G_2, \ldots, G_k$, possibly with overlapping node and edge set, however without any PII or correlation among node labels of different networks. In particular, let $u \in V_1$, $v \in V_2$, then $u = v$ does not imply that $x_{G_1}(u) = x_{G_2}(v)$. Thus, note that each network has its own function y_G. Assume y_{G_i} is given for all $i = 1, \ldots, k$ networks.

Problem Statement: Given the set of anonymized social networks $G_1, G_2, \ldots, G_k$ and the duplicate identification function for each network, y_G, learn a model that can evaluate the function $y_{G'}$ on *any* anonymized social network G'.

Note that since the networks are anonymized and there is no information on the labels across different networks, the model must leverage the structure of the network, as no other information is available. Moreover, note that k is usually small since duplicate information is often not available. This poses a difficult classification task that we tackle with a novel framework that we next describe.

6.2 Framework

Recall the goal of determining if a node of an anonymized social network has a duplicate. Since the focus is on the node, `RSM2NN` will take a local approach (which will also help its computational performance). In particular, only the network structure around the node of interest will be leveraged to determine if the node has a duplicate. Thus, the proposed framework is node centered and consists of three core steps:

1. Extract induced and connected subgraphs from G rooted at the node. Extract features concerning the nodes of this subgraph (e.g., its degree on G).

2. Match an extracted rooted subgraph from G to a reference graph. This is required to align the features across different instances of the same subgraph as well as identify the equivalence classes.

3. Aggregate the features across all extracted rooted subgraphs that are isomorphic, generating a single feature vector for each reference graph. The set of such vectors (across all reference graphs) is taken as the structural feature for the node.

Figure 6.2 illustrates the steps of `RSM2NN` . The first step is to extract rooted subgraphs of node 1, then match theses subgraphs with reference graph identifying the equivalence classes, then extract features according to the nodes in the different equivalence classes of the reference graph, and finally aggregate statistics across the multiple isomorphic subgraphs. In what follows these steps are described in detail.

At the end of these three steps, a node will have a feature vector that consists of multiple values for each possible reference graph. This information is then used to train a two-phase neural network model soon to be described. We first provide details of each step above.

6.2.1 Reference graphs

A fundamental aspect of the framework is the set of *reference graphs* as they drive the structural information that will be extracted from the network. A reference

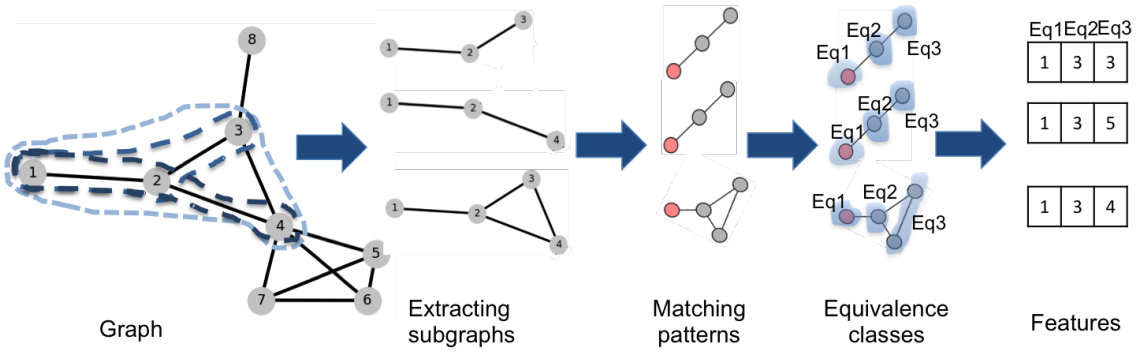


Figure 6.2: Framework

graph is a rooted connected small graph without node labels (except for the root). Figure 6.3 shows all possible reference graphs with three and four nodes. Since reference graphs are rooted, note that graph 7 is different from graph 8 in Figure 6.3. Although graphs 7 and 8 are isomorphic, the position of the root node makes them different. Let $\mathcal{R}$ denote the set of reference graphs under consideration. In this work, $\mathcal{R}$ is given by the set of all rooted connected graphs with 3, 4 or 5 nodes. In particular, $|\mathcal{R}| = 72$ in this case.

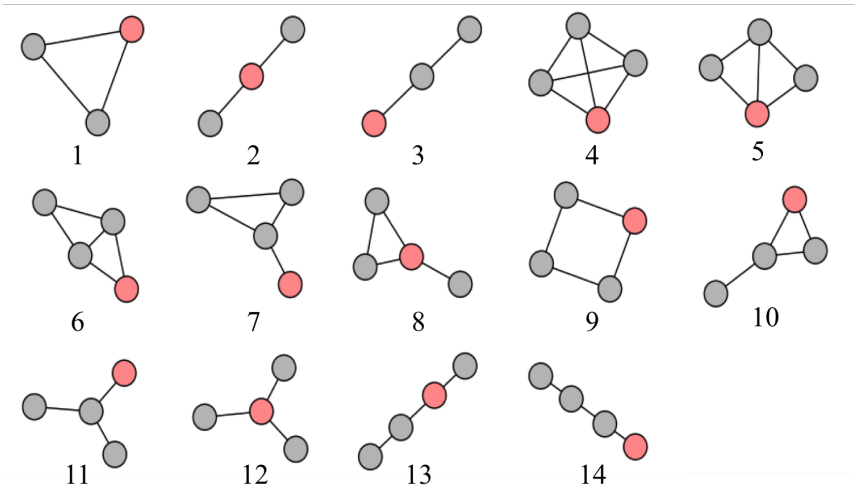


Figure 6.3: All possible reference graphs with three and four nodes.

Note that extracted subgraphs will have to be matched against a reference graph. Since node labels do not matter (except for the root), a *canonical labelling* of the graph will be used to establish the matching. Thus, for every reference graph we first perform a canonical labeling that labels the nodes uniquely and consistently, with the root node always identified as node 1. The canonical labelling cannot distinguish between nodes that belong to the equivalence class under automorphism. Thus, if there exists an automorphism between two (or more) nodes in a reference graph, their labels could be reversed in two different instances of this graph. Therefore, nodes cannot be reliably aligned based only on their canonical labels. For example,

in reference graph 8 (in Figure 6.3) there exists an automorphism between the two nodes with degree 2

This misalignment problem is avoided by considering the equivalence classes under automorphism and not nodes individually. Thus, for each reference graph $R \in \mathcal{R}$, we determine its set of equivalence classes $\mathcal{Q}_R$ along with the canonical labeling of its nodes, given by $\mathcal{L}_R$. Note that the root node has its own equivalence class, as this node is always identified (and labelled as 1 by the canonical labeling). However, the equivalence classes will strongly depend on the reference graph. For example, the canonical labelling of reference graph 8 gives label 1 to the root, label 2 to the node with degree 1, and labels 3 and 4 to the nodes with degree 2. Thus, it has the following equivalence classes $\mathcal{Q}_8 = \{\{1\}, \{2\}, \{3, 4\}\}$, while reference graph 4 has only two equivalence classes $\mathcal{Q}_4 = \{\{1\}, \{2, 3, 4\}\}$. Note that the minimum and maximum number of equivalence classes for a reference graph $R \in \mathcal{R}$ is 2 and 5, respectively. We use a very efficient canonical labelling algorithm known as *bliss* that also provides the equivalence classes [48].

Finally, the features extracted from a reference graph observed in the data will be associated to its equivalence classes, and not its nodes. Thus, equivalence classes is thee basic unit for feature identification, as we soon discuss.

6.2.2 Extracting subgraphs

The next step in the framework is to extract rooted induced subgraphs from the anonymized social network. Let $G = (V, E)$ denote the social network and consider a node $v \in V$ denoted as the root. An instance of a given reference graph is an induced subgraph of G rooted at v that is isomorphic to the reference graph (with the root corresponding to v). Since only rooted induced subgraphs are considered, it can be identified by the root and the node set, as edges are given by E.

Algorithm 4 shows the iterative strategy to extract all induced subgraphs rooted at v that correspond to a reference graph. The main idea of the algorithm is to use a subgraph of size i to construct subgraphs of size $i+1$ by considering every possible neighbor of a node in the smaller subgraph. The algorithm is iterative and starts with all edges (subgraphs of size 2) incident on v. For each edge, all subgraphs of size 3 that include that edge are constructed. Then, for each subgraph of size 3, all subgraphs of size 4 that include this subgraph of size 3 are constructed, and so on. The procedure AUGMENTSUBGRAPH is called to construct all subgraphs that can be constructed by adding a single node to a given subgraph. Note that the L_i contain all subgraphs of size i, for $i = 3, 4, 5$, and an element of L_i is a node set (without v). Finally, the parameter k is used to sample the subgraphs as opposed to considering all subgraphs. In particular, k denotes the number of subgraphs of

size $i+1$ that will be generated from a given subgraph of size i. Moreover, given the subgraph of size i, these k subgraphs are chosen uniformly at random among all possible subgraphs of size $i+1$.

Note that Algorithm 4 is executed for every node $v \in V$ in the anonymized social network G. Note that each subgraph rooted at node v will give rise to structural features that will characterize v, as we soon discuss.

Algorithm 4: Algorithm to extract rooted subgraphs.

```
Procedure subgraphsExtraction(v, k):
    L_2 ← AUGMENTSUBGRAPH(v, degree(v));
    for i ∈ (2, 3, 4) do
        L_i ← ∅; ;                                   // L_i is a set of sets
    for i ∈ (2, 3, 4) do
        for S ∈ L_i do
            L_{i+1} ← L_{i+1} ∪ AUGMENTSUBGRAPH(S, k);
    return L_3, L_4, L_5
Procedure augmentSubgraph(S,k):
    U ← ∅;
    for i ∈ S do
        U ← U ∪ {N(i)} ;                    // N(i) denotes neighbors of i
    U ← U \ S ;                            // Only consider new nodes in U
    if |U| > k then
        U ← randomly select k elements from U
    P ← ∅ ;                                          // P is a set of sets
    for i ∈ U do
        S' ← S ∪ {i};
        P ← P ∪ {S'};
    return P
```

6.2.3 Matching subgraphs with reference graphs

The extracted rooted subgraphs of the previous step must be matched to a reference graph. Moreover, the nodes of the subgraph must be aligned to a reference graph such that the equivalence classes can also be identified. This matching and node alignment (between a rooted subgraph and a reference graph) requires solving a rooted graph isomorphism problem. However, since the set $\mathcal{R}$ is small and every reference graph is also small (at most five nodes), this computation can be performed efficiently, as illustrated in Algorithm 5.

In particular, let $\mathcal{R}_{k,l}$ denote the set of reference graphs with k nodes and l edges, and note that it defines a partition of the set $\mathcal{R}$. Thus, $|\mathcal{R}_{k,l}|$ is rather small for any k and l and in particular is always smaller than 8. Recall that $\mathcal{L}_R$ is the

pre-computed canonical label for reference graph R. Thus, only the canonical label of the identified subgraph must be computed during matching, denoted by L' in the algorithm.

Algorithm 5: Algorithm to match subgraphs to reference graphs

```
Procedure matchingSubgraph(S):
    E' ← {(u, v) : u, v ∈ S ∧ (u, v) ∈ E};
    L' ← CANONICALLABEL(G' = (S, E'));
    k ← |S|;
    l ← |E'|;
    foreach R ∈ R_{k,l} do
        if L' == L_R then
            return n_R ;                    // id of reference graph R
```

6.2.4 Extracting and aligning features

The subgraph S is used to extract features for this subgraph from the original graph G. Recall that in order to be properly aligned, features must be associated to equivalence classes of this subgraph (that has been matched to a reference graph). Although any structural feature concerning the nodes of this subgraph can be extracted, this work considers the following features:

- degree and local clustering coefficient [45] of the root node;
- for each equivalence class, the minimum, maximum and average node degree within the nodes in the equivalence class;
- for each equivalence class, the minimum, maximum and average Adamic-Adar similarity coefficient [45] between the root node and each node in the equivalence class.

Note that the number of features generated by a subgraph depends on its number of equivalence classes. Moreover, the number of nodes in a given equivalence class will also vary, but the minimum, maximum and average is taken within this set (even if the size of the equivalence class is one). For example, since reference graph 8 has equivalence classes $\mathcal{Q}_8 = \{\{1\}, \{2\}, \{3, 4\}\}$, it will have $2 + 6 + 6 = 14$ features.

6.2.5 Feature aggregation

Note that a given reference graph may appear several times as different rooted subgraphs of a node v. Recall that each of these subgraph will have its features extracted from G. Since these features can vary significantly across these isomorphic

subgraphs, we summarize them using their average. To be sure, let $H_i(v)$ denote all the subgraphs rooted at v that correspond to reference graph i. Let $F_i(v)$ denote the average feature vector of node v for reference graph i, which is determined as the average across all feature vectors in $H_i(v)$. This feature vector along with the number of subgraphs considered, namely $|H_i(v)|$, is the feature associated with reference graph i for node v. Thus, every node in the network will have a vector $F_i(v)$ along with $|H_i(v)|$ for each reference graph $i \in R$. This final feature vector has information concerning all 72 reference graphs (size of R) and has dimension 1350.

6.2.6 Two-level neural network

Recall that the main goal is to classify nodes in a social network with respect to their duplicity. In particular, a binary classifier receives as input structural features for a node and outputs one if the node has a duplicate in the same network.

We propose a novel two-level neural network as the classification model. In the first level, each reference graph is considered separately, and has its own neural network and a single output. This first level leverages the potential of each reference graph in determining node duplicity. The second level is a different neural network that takes as input the output of the neural networks corresponding to the reference graphs. Thus, this second level leverages the potential of the set of reference graphs in determining node duplicity. Figure 6.4 show the architecture of the proposed neural network. Note that there are 72 neural networks in the first level, corresponding to the 72 reference graphs. The input to each of these networks is the feature vector $F_i(v)$ of node v along $|H_i(v)|$ and its dimension depends on the number of equivalence classes of reference graph i.

The neural networks associated with each reference graph, and also the neural network in the second level, all have the same structure, formed by two hidden layers each with 20 neurons and a single output neuron with a Sigmoid activation function. The Relu activation function is used within the network. The choice of 2 and 20 is assessed in Section 6.3.

Note that the input to the model are features of a node from a social network, but the social network is not required. Thus, nodes of a network can be classified in a trained model even if the model has never seen any node from that network. This is possible because only structural features that are local to the node are considered, and not label information or global network information (i.e., number of nodes in the network).

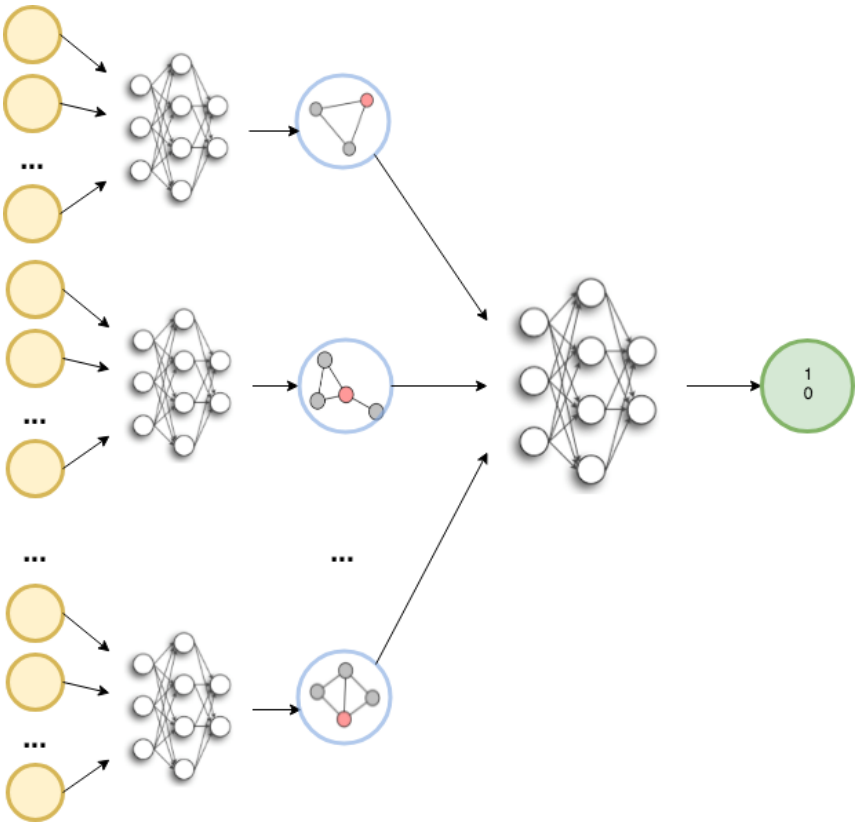


Figure 6.4: The proposed two-level neural network: while the first level considers references graphs separately to identify duplicates, the second level integrates the output of the reference graphs.

6.3 Experiments and results

We evaluate our approach using different experiments and datasets focusing on the following questions: 1) Inductive learning: Model trained using nodes from one network is used to classify nodes of different networks; 2) Training size trade off: performance improvement when training with nodes of multiple networks; 3) Impact of subgraph sampling, and 4) Comparison with other existing approaches.

6.3.1 Datasets

A challenge in duplicate identification in social networks is the lack of publicly available datasets with ground truth information, in particular when considering multiple networks. To overcome this problem, a relatively large dataset was collected in the context of this work. Moreover, three other datasets were used in the evaluation:

- Google Scholar (GS)[1]: provides a wide range of bibliographic information from the academic literature. In particular, GS maintains a profile for each individual where its publications are listed. We randomly select 3000 researchers

[1]Link to Google Scholar website: https://scholar.google.com/

	DBLP	Google Scholar	NIH	Wikidata
Number of networks	1	3000	162	255
Nodes per network	1827903	56.7 (28.4)	71.7 (12.3)	21.1 (24.0)
Smallest network (nodes)	-	10	39	10
Largest network (nodes)	-	227	103	188
Fraction of ambiguous nodes	0.02	0.07 (0.04)	0.6 (0.1)	0.83 (0.16)
Edges per network	8643727	140.6 (85.2)	449.1 (158.5)	38.3 (76.6)

Table 6.1: Characterizing datasets.

that have a Google Scholar profile. Each individual yields a ego-centered collaboration network, formed by all names that appear in the author list of all publications listed in the profile (edges indicate that two names appear in the same publication at least once).

- DBLP[2]: contains bibliographic information mostly for publications in Computer Science and its database is publicly available for download (this work used the May 2014 version). While some individuals do appear with different names in distinct publication records, DBLP maintains a record for a known set of such individuals[3], which is taken as the ground truth for synonyms. Note that a single collaboration network is constructed where names as it appears in the author list of the publications correspond to nodes.

- NIH[4]: a clinical dataset provided by the National Institutes of Health (NIH) and used in [1]. This dataset has 162 networks and records family medical histories.

- Wikidata[5]: a public dataset crawled from the structured knowledge repository Wikidata and used in [1]. The dataset has 419 networks but only 255 have been considered as some networks are either too small or do not have duplicate or non-duplicate nodes. Only networks with ten or more nodes that have both duplicate and non-duplicate nodes have been considered.

Table 6.1 shows the number of networks for each dataset and the average number of nodes and edges per network. Google Scholar, NIH and Wikidata are formed by many networks. Whole DBLP dataset has a single network. Note that each network has duplicate and non-duplicate nodes. The average number of nodes per network and the fraction of duplicate nodes vary significantly across the datasets allowing for a more thorough evaluation of the proposed framework.

[2]Link to DBLP website: https://dblp.uni-trier.de/

[3]See details on DBLP's handling of synonyms at http://dblp.uni-trier.de/faq/How_does_dblp_-handle_ homonyms_and_synonyms.html

[4]Link to NIH: https://www.nih.gov/

[5]Link to Wikidata: https://www.wikidata.org

6.3.2 Metrics

The metrics used to evaluate the performance are:

- Average Precision (AP): the precision-recall curve yields precision-recall pairs for different classification threshold values. The average precision summarizes the precision-recall curve as the weighted average of precision values at each threshold where the increase in recall from the previous threshold is used as the weight [49].

- Area Under ROC Curve (AUC): the ROC curve represent the false positive ratio and true positive ratio for different classification threshold values. The area under this curve (AUC) measures the average performance of the classifier [49].

It is important to emphasize that most of the datasets have imbalanced classes with a large number of negative examples. Thus, if all examples are classified as negative (non ambiguous), the accuracy would be high but that does not reflect the performance of the framework in identifying duplicate nodes. Thus, precision and AP are more adequate metrics in this scenario.

6.3.3 Inductive learning

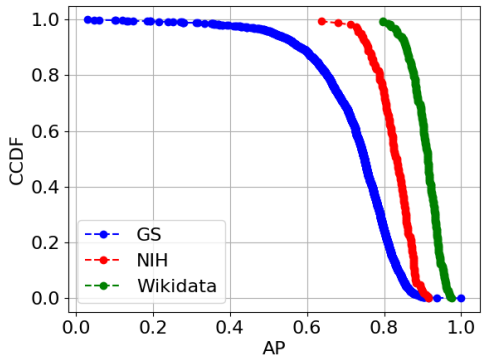
Can nodes from a single network of the dataset be used to train a model to classify nodes of another network in the same dataset?

We take one network of a dataset and use all its nodes as examples of duplicate or non-duplicate nodes to train `RSM2NN` . Then, we apply the model to classify all the nodes of another network. This provides the classification performance for this network. We repeat this classification for 100 test networks, each providing the performance of classifying all nodes of the network. We repeat this experiment several times considering different networks to train the model (chosen randomly).

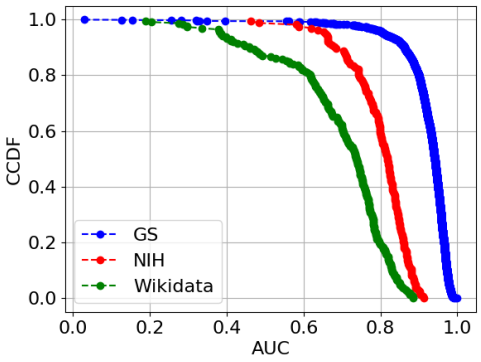
Figure 6.5 shows the Complementary Cumulative Distribution Function (CCDF) of AP and AUC for the three different datasets. Figure 6.5a shows that around 60% of the test networks for GS have AP greater than 70% and for NIH and Wikidata around 60% of the test networks have AP greater than 80%. Figure 6.5b shows AUC and for GS around 70% of the test networks are greater than 90% and for NIH and Wikidata are greater than 70%. Note that the AP curve for Google Scholar drops faster than for NIH and Wikidata and this behaviour is the opposite for AUC. This happens because the fraction of duplicate nodes is much smaller in GS than in the other datasets.

Next, we evaluate the performance eight networks are used to train, Figure 6.5c and 6.5d, the performance improves significantly. For NIH and Wikidata 40% of the

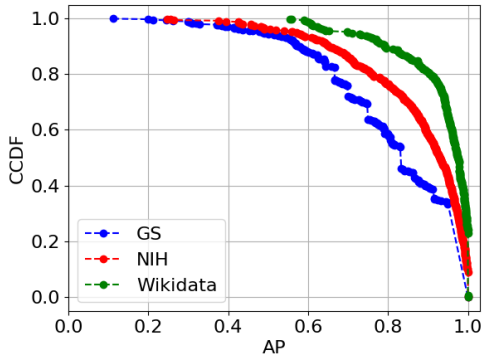
networks have AP larger than 95%, for Google Scholar this value is around 90%, Figure 6.5c. The AUC for Google Scholar for 60% of the networks are larger than 95% and for NIH and Wikidata are more than 80%, Figure 6.5d. Of course, the performance strongly depends on the train and test networks. Intuitively, training on more networks allows for a more robust classifier since the model will see different kind of nodes.



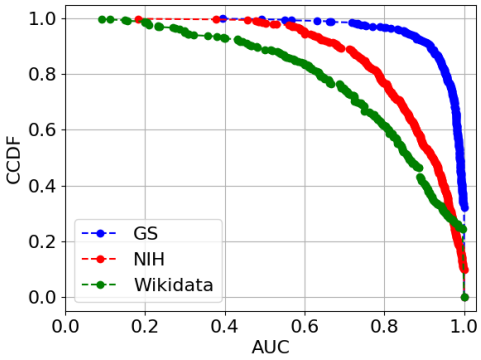
(a) Training size of 1 network



(b) Training size of 1 network



(c) Training size of 8 networks



(d) Training size of 8 networks

Figure 6.5: Distribution of the performance of `RSM2NN` using 1 or 8 networks to train the model and 100 different networks from the same dataset to test.

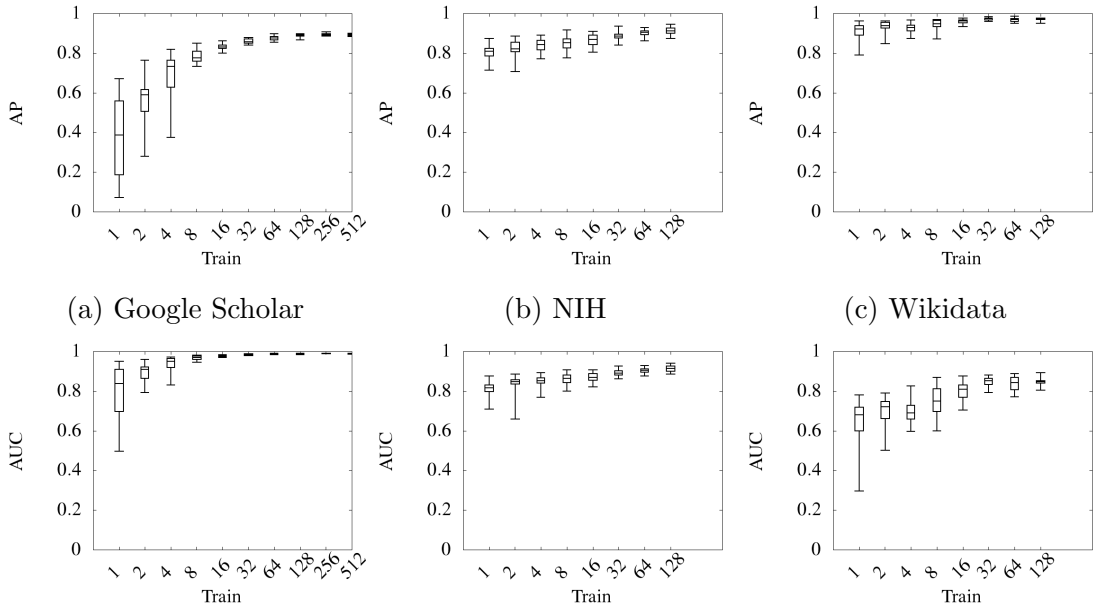
6.3.4 Training size trade off

Section 6.3.3 evaluated the performance from the perspective of networks as all nodes used in training and testing were from the same network. In this section we evaluate `RSM2NN` from the perspective of nodes. We train the network with nodes from different networks and test with nodes from many different networks.

To run this experiment we use 1, 2, 4, ..., 128 networks to train the classifier and test with nodes from 20% of the networks in the same dataset. For each scenario the experiment is repeated ten times.

The results are shown in Figure 6.6. For all datasets both AP and AUC increase

on average as more networks are used for training, and the gain depends on the dataset and the metric. For Google Scholar, with sixteen networks we obtain an AP higher than 80% and a AUC higher than 95%. For NIH and Wikidata an AP higher than 80% on average is obtained with only one network. The average AP for Google Scholar grows but not to 100% and in Wikidata it starts high and goes to 100%. The opposite behavior happens with AUC. This is because the datasets are very different, GS has an average of 7% duplicate nodes in a network while Wikidata is around 80%.



(a) Google Scholar (b) NIH (c) Wikidata

(d) Google Scholar (e) NIH (f) Wikidata

Figure 6.6: AP and AUC of classifier when varing the amount of the networks used to train the classifier.

6.3.5 Impact of sampling

The DBLP dataset is composed of a single large network and its nodes have a rich neighbourhood making it very expensive to compute all induced subgraphs. An alternative is to sample the sugraphs. In this section we verify the performance when using only a small sample of the subgraphs rooted at a node to generate its features.

The subgraphs of nodes in DBLP have been sampled with $k = 20$ as given by in Algorithm 1. To run these experiments 100000 nodes were selected and 80% of them used for training and 20% for testing. The experiment was repeated 100 times. Table 6.2 shows the results, the average AP is 76% and the average recall 88%.

For Google Scholar we generate all sugraphs and also a sample using Algorithm1, with $k = 20$. Thus, we are able to compare the results of the framework when using

all subgraphs or a small sample. To run these experiments we use nodes from 80% of the networks for training and nodes from 20% of the networks for testing, considering the ensemble of all nodes, without considering the networks separately. Table 6.2 shows the results with AUC of 99% when all subgraphs are used and when a sample is used. The average AP when using a sample of the subgraphs is 89% and is 1% better then when considering all subgraphs, recall is 74% and is 9% better. These results indicate that it is possible to use a sample of the subgraphs without loosing performance, and some times even improving.

Datasets	**Subgraphs extraction**	**AUC**	**AP**	**Precision**	**Recall**	**F1**
GS	all	0.990(0.001)	0.870(0.005)	0.905(0.022)	0.655(0.024)	0.759(0.010)
GS	sampled ($k = 20$)	0.990(0.002)	0.898(0.010)	0.890(0.022)	0.743(0.017)	0.809(0.011)
DBLP	sampled ($k = 20$)	0.750(0.005)	0.763(0.008)	0.602(0.009)	0.884(0.013)	0.716(0.003)

Table 6.2: Performance of the framework when using a sample of the subgraphs.

6.3.6 Hidden layers and neurons

Recall that `RSM2NN` has two hyper parameters for the neural network: the number of hidden layers and the number of neurons per layer in the neural network for each neural network in the two levels.

Table 6.3 shows the performance for different hyperparameter values. These experiments considered the GS dataset and 80% of the networks were used to train `RSM2NN` and 20% to test. As indicated, performance is better when considering 20 neurons instead of 100. Results are similar when considering one or two hidden layers, so we adopt two.

6.3.7 Baseline comparison methods

To validate the performance of the proposed framework, we compare it against other features and methods. In particular, we compare `RSM2NN` with methods that generate features from the network and also with other classifiers using the same features of `RSM2NN` .

node2vec [50]: This framework learns features in a low-dimensional space for nodes that maximizes the likelihood of preserving network neighborhoods.

SVM [40]: Support Vector Machine (SVM) is a rule established classifier defined by a separating hyperplane using labeled training data.

NN [51]: compare `RSM2NN` with a simple neural network with two layers and twenty neurons each (same parameters adopted in `RSM2NN` but no two level hierarchy).

Neural Network Parameters		**Metrics**				
hidden layers	neurons	AUC	AP	Precision	Recall	F1
1	20	0.992(0.001)	0.910(0.007)	0.925(0.019)	0.728(0.024)	0.814(0.011)
	100	0.987(0.003)	0.887(0.009)	0.903(0.045)	0.703(0.035)	0.788(0.009)
2	20	0.990(0.002)	0.898(0.010)	0.923(0.022)	0.710(0.025)	0.802(0.012)
	100	0.963(0.012)	0.826(0.021)	0.910(0.035)	0.626(0.037)	0.740(0.020)
4	20	0.986(0.001)	0.878(0.009)	0.955(0.031)	0.644(0.021)	0.769(0.011)
	100	0.935(0.001)	0.755(0.002)	0.917(0.028)	0.561(0.030)	0.696(0.011)

Table 6.3: Classifier parameters and its performance.

PSL [1]: a model that incorporates statistical signals, such as name similarity, relational information and logical constraints and showed how to integrate these features using probabilistic soft logic. This methods used PII such as names and relationships between individuals.

Table 6.4 shows the results for the GS and DBLP datasets. The performance of `RSM2NN` is superior to all others in both datasets. When using node2vec to generate node features, the performance using SVM or using a simple Neural Network is lower than when using the features of `RSM2NN` with these same classifiers. Node2vec places nodes that are neighbors near each other in feature space and since duplicate and non-duplicate nodes are neighbors, the classification algorithm cannot easily distinguish them. When using the features generated by `RSM2NN` , SVM and the simple Neural Network outperform node2vec but are inferior to the `RSM2NN` framework, in both datasets. This indicates that both the features generated by `RSM2NN` and the two-level neural network architecture are important for the task at hand.

Table 6.5 compares the results of `RSM2NN` with the method proposed by Kouki et al. [1]. Considering recall `RSM2NN` performed equal or better while the precision for `RSM2NN` in both datasets was higher than 84% (the other method was 91%). It is important to note that our method uses only structural information, while in [1] Kouki et al. leverages PII training and classification such as names and personal relationships. Thus, it is quite remarkable that results are similar which indicates the potential of leveraging structural information.

6.4 Conclusion

In this chapter we proposed a framework to identify duplicates for the synonym problem. The framework involves creating node features based only on the structure of the network, and thus, considers an anonymous network. These features are generated by rooted subgraphs of size 3, 4 and 5 and is used into a two level neural network. This specific neural network considers in the first level each reference graph separately. Then, the output of these neural network are concatenated to the second level of neural network that combines it to a single output that classifies the node as duplicate or non-duplicate.

The experiments with four different real datasets, DBLP, Google Scholar, NIH and Wikidata, indicated that `RSM2NN` can effectively classify duplicate nodes of a network it has never seen before, when trained with nodes of other networks in the same dataset. The number of networks used to train increases performance. With only sixteen networks an AUC higher than 80% for all datasets is achieved. As computing all subgraphs of some networks is very expensive, we compared the results using a sample and the result remain competitive. When we compare `RSM2NN`

Dataset	Method	AUC	AP	Precision	Recall	F1
DBLP	node2vec+SVM	0.500(0.000)	0.033(0.001)	-	-	-
	node2vec+NN	0.519(0.012)	0.058(0.031)	0.162(0.079)	0.051(0.032)	0.077(0.046)
	`RSM2NN` Features+SVM	0.541(0.005)	0.571(0.009)	0.572(0.001)	0.976(0.010)	0.721(0.008)
	`RSM2NN` Features+NN	0.560(0.006)	0.555(0.024)	0.581(0.025)	0.563(0.011)	0.572(0.018)
	`RSM2NN`	0.750(0.005)	0.763(0.008)	0.602(0.009)	0.884(0.013)	0.716(0.003)
Google Scholar	node2vec+SVM	0.500(0.000)	0.046(0.001)	-	-	-
	node2vec+NN	0.504(0.002)	0.048(0.002)	0.352(0.090)	0.009(0.005)	0.018(0.010)
	`RSM2NN` Features+SVM	0.539(0.008)	0.124(0.003)	0.951(0.007)	0.078(0.001)	0.145(0.009)
	`RSM2NN` Features+NN	0.720(0.023)	0.254(0.021)	0.482(0.015)	0.469(0.050)	0.474(0.024)
	`RSM2NN`	0.990(0.001)	0.870(0.005)	0.905(0.022)	0.655(0.024)	0.759(0.010)

Table 6.4: Comparing `RSM2NN` with noded2vec, SVM and a standard Neural Network. The - in the SVM algorithm did not classify any node as ambiguous for experiment with node2vec features for both datasets, that is the reason there is no result for them

	NIH			Wikidata		
Method	**Precision**	**Recall**	**F-measure**	**Precision**	**Recall**	**F-measure**
ICDM2017 (Relational Info (1^{st} degree)	0.956 (0.006)	0.924 (0.024)	0.940 (0.014)	0.914 (0.016)	0.880 (0.018)	0.896 (0.006)
ICDM2017 (Relational Info (1^{st} + 2^{nd} degree)	0.961 (0.011)	0.931 (0.020)	0.946 (0.010)	-	-	-
`RSM2NN` (Relational Info 1^{st} degree)	0.842 (0.037)	0.925 (0.026)	0.881 (0.020)	0.896 (0.018)	0.963 (0.015)	0.928 (0.011)
`RSM2NN` (Relational Info 1^{st} + 2^{nd} degree)	0.850 (0.025)	0.923 (0.028)	0.884 (0.016)	-	-	-

Table 6.5: Performance of `RSM2NN` and baseline classifiers. Numbers in parenthesis indicate standard deviations. This table shows the metrics used in [1] to evaluate its performance.

with baseline methods we obtain better results.

The proposed framework can be used to tackle other classification tasks on anonymous social networks. In particular, with respect to name ambiguity, a more widely studied variation is when a single name refers to more than one individual. The goal here would be to determine if a node in the anonymous social network refers to multiple individuals.

Chapter 7

Conclusions

This thesis considered the problem of ambiguity in anonymous social networks where duplicated references (network nodes) correspond to the same person, the synonym problem.

First we studied the reasons for individuals to appear with multiple names and provided a characterization of these name appearance over time for different authors. The individuals with duplicated names were classified into one of three classes: rare, representing the cases where individuals use one name most of the time with the second name appearing in just a few publications; swap, representing individuals that start using one name but changes to another name; and co-appearance, for individuals that regularly two use or more names. We could observe that classes in the DBLP and Google Scholar datasets, and the network structure for individuals of each class are different. No related work has looked into this problem before, and we believe that identification of root causes is a fundamental step towards for better understanding name ambiguity.

We also proposed a probabilistic model to introduce synonyms in a social network using three parameters. Parameter p is the probability to duplicate a vertex, q is the probability to copy an edge between a neighbour of the original vertex and its duplicated version and r is the probability to delete an edge between the original vertex and the neighbour. When varying these parameters the network structure changes significantly and we analyze how these differences affects a naive algorithm that finds the synonym in the network.

Three algorithms were proposed to solve the problem of ambiguity in social networks using anonymous social networks and no other context information beyond the network structure. The first one is a simple algorithm that considers the whole network and identifies the synonym nodes based only on heuristics, such as the distance between the nodes and the number of common neighbors. It was applied to networks generated by two models of social networks (Erdos-Renyi and Watts-Strogatz) after introducing ambiguity using the model previously proposed. For

both network models the algorithm showed differences when finding the ambiguous nodes with the variation of parameter p. The most sensitive parameter for the performance of the algorithm is r, the precision is more than 90% for values of r lower than 0.5, independent of the other parameters p and q. As r grows the precision and recall decrease as more original edges are removed and the algorithm fails to find the original vertex that corresponds to the duplicated one.

The second algorithm considers an ego-centered ambiguous network, a network that has a profile owner, and the output is the set of nodes that correspond to that individual. The algorithm is based on the problems of independent set and dominating set and consider special characteristics of the synonym ambiguity problem. The algorithm was evaluated using two real bibliographic datasets, DBLP and Google Scholar, and for all datasets the precision is more than 77% on average and for Google Scholar dataset the precision is around 97%. This algorithm presented a very high success rate in solving this name ambiguity.

The last algorithm is a framework that determines if a node of a network has a duplicate in the same network. First the framework extracts induced subgraphs from a graph rooted at a node and extract features concerning the nodes of this subgraph. The second step is to match the subgraphs to a reference graph in order to align the features across different instances of the same subgraph. Then, the features are aggregated across all subgraphs that are isomorphic, generating a single feature vector for the node. This vector is the input of a two-level neural network that we proposed specifically to tackle the problem. The first level is a neural network for each subgraph pattern and the output is concatenated to neural network in a second level. The output of the second level classifies a node as having a duplicate. We evaluated this framework using two collaborations networks, DBLP and Google Scholar, and two familial networks, NIH and Wikidata. The framework was able to identify duplicates with a median AUC of 97% and 83% when trained with nodes from a *single* network of the same domain. In comparison to other approaches (node features or classification models) it is significantly superior and also competitive with recent methods that leverage contextual information.

All the proposed algorithms have as input an anonymous social network (no label information is used) and considers only network structure to find nodes that are synonyms. Experimental results with real available datasets indicate that is possible to obtain a good performance in identifying the synonym nodes considering only the network structure.

7.1 Future work

Below follows some activities to continue and enhance the work already done in this thesis.

In chapter 4 we proposed a model to introduce ambiguity in a network. This model is limited because it duplicates nodes only once, so the network does not have more than two nodes for the same person, which does not represent reality. The model also treats every node identically and does not reflect the different reasons that give rise to node ambiguity, as seen in Chapter 3. An improvement could leverage the knowledge of the root causes for synonyms discussed in Chapter 3 and consider different mechanisms to generate node duplication and edges in accordance to our empirical finding, where ambiguous nodes can be very different. An enhanced model, can be compared to real ambiguous networks to measure its effectiveness in representing network ambiguity.

Another future work could be the improvement of the algorithm proposed in chapter 5 which starts by selecting the node with highest degree. If the algorithm makes a wrong first choice all the following steps lead to wrong conclusions and it will not succeed. An improvement could be to study other criteria to start the algorithm and even make it possible to rollback and restart.

In chapter 6 a proposed framework that uses neural network to classify a node into ambiguous or not is applied to the synonym ambiguity problem and we believe that it can be used to tackle other classification tasks on anonymous social networks. In particular, with respect to name ambiguity, a more widely studied variation is when a single name refers to more than one individual (homonym problem). The same framework can be applied to determine if a node in the anonymous social network refers to multiple individuals.

Another application of the framework proposed in chapter 6 could be its use to classify the nodes of an anonymous network in arbitrary number of classes. The algorithm could be used to classify nodes into different classes, not just two, without making any restriction on the network.

Bibliography

[1] KOUKI, P., PUJARA, J., MARCUM, C., et al. "Collective Entity Resolution in Familial Networks". In: *2017 IEEE International Conference on Data Mining*, pp. 227–236, Nov 2017.

[2] FERREIRA, A. A., GONÇALVES, M. A., LAENDER, A. H. "A Brief Survey of Automatic Methods for Author Name Disambiguation", *SIGMOD Rec.*, v. 41, n. 2, pp. 15–26, ago. 2012.

[3] AMANCIO, D. R., OLIVEIRA JR, O. N., DA F. COSTA, L. "Topological-collaborative approach for disambiguating authors' names in collaborative networks", *Scientometrics*, v. 102, n. 1, pp. 465–485, 2015.

[4] HERMANSSON, L., KEROLA, T., JOHANSSON, F., et al. "Entity Disambiguation in Anonymized Graphs Using Graph Kernels". In: *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pp. 1037–1046, New York, NY, USA, 2013. ACM.

[5] SAHA, T. K., ZHANG, B., HASAN, M. A. "Name disambiguation from link data in a collaboration graph using temporal and topological features", *Social Network Analysis and Mining*, v. 5, n. 1, pp. 11, Mar 2015.

[6] ZHANG, B., AL HASAN, M. "Name Disambiguation in Anonymized Graphs Using Network Embedding". In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 1239–1248, New York, NY, USA, 2017. ACM.

[7] GETOOR, L., MACHANAVAJJHALA, A. "Entity Resolution: Theory, Practice, Open Challenges", *Proc. VLDB Endow.*, v. 5, n. 12, pp. 2018–2019, ago. 2012.

[8] BEKKERMAN, R., MCCALLUM, A. "Disambiguating Web Appearances of People in a Social Network". In: *Proceedings of the 14th International Conference on World Wide Web*, pp. 463–470, New York, NY, USA, 2005. ACM.

[9] VU, Q. M., MASADA, T., TAKASU, A., et al. "Using a Knowledge Base to Disambiguate Personal Name in Web Search Results". In: *Proceedings of the 2007 ACM Symposium on Applied Computing*, SAC '07, pp. 839–843, New York, NY, USA, 2007. ACM.

[10] DIEHL, C. P., GETOOR, L., NAMATA, G. "Name Reference Resolution in Organizational Email Archives". In: *Proceedings of the 2006 SIAM International Conference on Data Mining*, cap. 7, pp. 70–81, 2006.

[11] AL-MUBAID, H., CHEN, P. "Biomedical Term Disambiguation: An Application to Gene-Protein Name Disambiguation". In: *Third International Conference on Information Technology: New Generations*, pp. 606–612, April 2006.

[12] SMITH, D. A., CRANE, G. "Disambiguating Geographic Names in a Historical Digital Library". In: *Research and Advanced Technology for Digital Libraries: 5th European Conference, ECDL 2001 Darmstadt, Germany, September 4-9, 2001 Proceedings*, pp. 127–136, Springer Berlin Heidelberg, 2001.

[13] KIM, J. "Evaluating author name disambiguation for digital libraries: a case of DBLP", *Scientometrics*, v. 116, pp. 1867–1886, 2018.

[14] SCOVILLE, C. L., JOHNSON, E. D., MCCONNELL, A. L. "When A. Rose is not A. Rose: the vagaries of author searching", *Medical Reference Services Quaterly*, v. 22, n. 4, pp. 1–11, 2003.

[15] CULOTTA, A., KANANI, P., HALL, R., et al. "Author disambiguation using error-driven machine learning with a ranking loss function". In: *In Proceedings of AAAI Sixth International Workshop on Information Integration on the Web*, 2007.

[16] FERREIRA, A. A., VELOSO, A., GONÇALVES, M. A., et al. "Effective Self-training Author Name Disambiguation in Scholarly Digital Libraries". In: *Proceedings of the 10th Annual Joint Conference on Digital Libraries*, pp. 39–48, New York, NY, USA, 2010. ACM.

[17] FERREIRA, A. A., VELOSO, A., GONALVES, M. A., et al. "Self-training author name disambiguation for information scarce scenarios", *Journal of the Association for Information Science and Technology*, v. 65, n. 6, pp. 1257–1278, 2014.

[18] DE CARVALHO, A. P., FERREIRA, A. A., LAENDER, A. H. F., et al. "Incremental unsupervised name disambiguation in cleaned digital libraries", *Jounal of Information and Data Management*, v. 2, n. 3, pp. 289–304, 2011.

[19] FAN, X., WANG, J., PU, X., et al. "On Graph-Based Name Disambiguation", *J. Data and Information Quality*, v. 2, n. 2, pp. 10:1–10:23, fev. 2011.

[20] HAN, H., GILES, L., ZHA, H., et al. "Two Supervised Learning Approaches for Name Disambiguation in Author Citations". In: *Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, pp. 296–305, New York, NY, USA, 2004. ACM.

[21] HAN, H., XU, W., ZHA, H., et al. "A Hierarchical Naive Bayes Mixture Model for Name Disambiguation in Author Citations". In: *Proceedings of the 2005 ACM Symposium on Applied Computing*, pp. 1065–1069, New York, NY, USA, 2005. ACM.

[22] HAN, H., ZHA, H., GILES, C. L. "Name Disambiguation in Author Citations Using a K-way Spectral Clustering Method". In: *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, pp. 334–343, New York, NY, USA, 2005. ACM.

[23] LEVIN, F. H., HEUSER, C. A. "Evaluating the Use of Social Networks in Author Name Disambiguation in Digital Libraries", *Jounal of Information and Data Management*, v. 1, n. 2, pp. 183–197, 2010.

[24] SHIN, D., KIM, T., CHOI, J., et al. "Author name disambiguation using a graph model with node splitting and merging based on bibliographic information", *Scientometrics*, v. 100, n. 1, pp. 15–50, 2014.

[25] AMANCIO, D. R., JR., O. N. O., DA F. COSTA, L. "On the use of topological features and hierarchical characterization for disambiguating names in collaborative networks", *EPL (Europhysics Letters)*, v. 99, n. 4, pp. 48002, 2012.

[26] SANTANA, A. F., GONALVES, M. A., LAENDER, A. H. F., et al. "On the combination of domain-specific heuristics for author name disambiguation: the nearest cluster method", *International Journal on Digital Libraries*, v. 16, n. 3, pp. 229–246, 2015.

[27] KANANI, P., MCCALLUM, A., PAL, C. "Improving Author Coreference by Resource-bounded Information Gathering from the Web". In: *Proceedings of the 20th International Joint Conference on Artifical Intelligence*,

IJCAI'07, pp. 429–434, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[28] PEREIRA, D. A., RIBEIRO-NETO, B., ZIVIANI, N., et al. "Using Web Information for Author Name Disambiguation". In: *Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries*, pp. 49–58, New York, NY, USA, 2009. ACM.

[29] ON, B.-W., ELMACIOGLU, E., LEE, D., et al. "Improving Grouped-Entity Resolution Using Quasi-Cliques". In: *Proceedings of the Sixth International Conference on Data Mining*, pp. 1008–1015, Washington, DC, USA, 2006. IEEE Computer Society.

[30] ZHANG, B., SAHA, T. K., HASAN, M. A. "Name disambiguation from link data in a collaboration graph". In: *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on*, pp. 81–84, Aug 2014.

[31] TORVIK, V. I., WEEBER, M., SWANSON, D. R., et al. "A probabilistic similarity metric for medline records: A model for author name disambiguation", *Journal of the American Society for Information Science and Technology*, v. 56, pp. 40–158, 2005.

[32] LIU, Y., LI, W., HUANG, Z., et al. "A fast method based on multiple clustering for name disambiguation in bibliographic citations", *Journal of the Association for Information Science and Technology*, v. 66, n. 3, pp. 634–644, 2015.

[33] ZHANG, Y., ZHANG, F., YAO, P., et al. "Name Disambiguation in AMiner: Clustering, Maintenance, and Human in the Loop." In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1002–1011, New York, NY, USA, 2018. ACM.

[34] KIM, J., KIM, J. "The Impact of Imbalanced Training Data on Machine Learning for Author Name Disambiguation", *Scientometrics*, v. 117, n. 1, pp. 511–526, out. 2018.

[35] SANTANA, A. F., GONCALVES, M. A., LAENDER, A. H. F., et al. "Incremental author name disambiguation by exploiting domain-specific heuristics", *Journal of the Association for Information Science and Technology*, v. 68, n. 4, pp. 931–945, 2017.

[36] WANG, X., TANG, J., CHENG, H., et al. "ADANA: Active Name Disambiguation". In: *2011 IEEE 11th International Conference on Data Mining*, pp. 794–803, Dec 2011.

[37] KM, P., MONDAL, S., CHANDRA, J. "A Graph Combination With Edge Pruning-Based Approach for Author Name Disambiguation", *Journal of the Association for Information Science and Technology*, v. 0, n. 0, 2019.

[38] SHEN, W., WANG, J., HAN, J. "Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions", *IEEE Transactions on Knowledge and Data Engineering*, v. 27, n. 2, pp. 443–460, Feb 2015.

[39] HARTIGAN, J. A., WONG, M. A. "A K-Means Clustering Algorithm", *Applied Statistics*, v. 28, pp. 100–108, 1979.

[40] CORTES, C., VAPNIK, V. "Support-Vector Networks", *Mach. Learn.*, v. 20, n. 3, pp. 273–297, 1995.

[41] KIM, J., DIESNER, J. "Distortive effects of initial-based name disambiguation on measurements of large-scale coauthorship networks", *Journal of the Association for Information Science and Technology*, v. 67, n. 6, pp. 1446–1461, 2016.

[42] GOMIDE, J., KLING, H., FIGUEIREDO, D. "A Model for Ambiguation and an Algorithm for Disambiguation in Social Networks". In: *Complex Networks VI*, Studies in Comp. Intelligence, Springer, pp. 37–44, 2015.

[43] FEGLEY, B. D., TORVIK, V. I. "Has Large-Scale Named-Entity Network Analysis Been Resting on a Flawed Assumption?" *PLOS ONE*, v. 8, n. 7, pp. 1–16, 07 2013.

[44] KIM, J., DIESNER, J. "The effect of data pre-processing on understanding the evolution of collaboration networks", *Journal of Informetrics*, v. 9, n. 1, pp. 226 – 236, 2015.

[45] NEWMAN, M. *Networks: An Introduction*. New York, NY, USA, Oxford University Press, Inc., 2010.

[46] GAREY, M. R., JOHNSON, D. S. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA, W. H. Freeman & Co., 1990.

[47] KANG, I.-S., KIM, P., LEE, S., et al. "Construction of a large-scale test set for author disambiguation", *Information Processing & Management*, v. 47, n. 3, pp. 452 – 465, 2011.

[48] JUNTTILA, T., KASKI, P. "Engineering an efficient canonical labeling tool for large and sparse graphs". In: Applegate, D., Brodal, G. S., Panario, D., et al. (Eds.), *Proceedings of the Ninth Workshop on Algorithm Engineering and Experiments and the Fourth Workshop on Analytic Algorithms and Combinatorics*, pp. 135–149, 2007.

[49] KOHAVI, R., PROVOST, F. "Glossary of Terms", *Machine Learning*, v. 30, n. 2-3, pp. 271–274, 1998.

[50] GROVER, A., LESKOVEC, J. "Node2Vec: Scalable Feature Learning for Networks". In: *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864, New York, NY, USA, 2016. ACM.

[51] ALPAYDIN, E. *Introduction to Machine Learning*. The MIT Press, 2010.