COLLECTIVE CLASSIFICATION OF USER EMOTIONS IN TWITTER

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

IBRAHIM ILERI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
COMPUTER ENGINEERING

AUGUST 2015

Approval of the thesis:

COLLECTIVE CLASSIFICATION OF USER EMOTIONS IN TWITTER

submitted by **IBRAHIM ILERI** in partial fulfillment of the requirements for the degree of **Master of Science in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Adnan Yazıcı _____
Head of Department, **Computer Engineering**

Assoc. Prof. Dr. Pınar Karagöz _____
Supervisor, **Computer Engineering Department, METU**

Examining Committee Members:

Prof. Dr. Ahmet Coşar
Computer Engineering Department, METU _____

Assoc. Prof. Dr. Pınar Karagöz
Computer Engineering Department, METU _____

Prof. Dr. Nihan Kesim Çiçekli
Computer Engineering Department, METU _____

Prof. Dr. İsmail Hakkı Toroslu
Computer Engineering Department, METU _____

Assist. Prof. Dr. Alev Mutlu
Computer Engineering Department, Kocaeli University _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: IBRAHIM ILERI

Signature :

ABSTRACT

COLLECTIVE CLASSIFICATION OF USER EMOTIONS IN TWITTER

Ileri, Ibrahim
M.S., Department of Computer Engineering
Supervisor : Assoc. Prof. Dr. Pınar Karagöz

August 2015, 75 pages

The recent explosion of social networks has generated a big amount of data including user opinions about varied subjects. For classifying the sentiment of user postings, many text-based techniques have been proposed in the literature. As a continuation of sentiment analysis, there are also studies on the emotion analysis. Because of the fact that many different emotions are needed to be dealt with at this point, the problem becomes much more complicated. In this thesis, a different user-centric approach is considered that connected users may be more likely to hold similar emotions; therefore, leveraging relationship information can complement user-level sentiment inference task in social networks. Employing Twitter as a source for experimental data and working with a proposed collective classification algorithm, users whose emotions are not known on subject, are predicted in an effective and collaborative setting.

Keywords: Social Networks, Sentiment Analysis, Collective Classification

ÖZ

TWİTTER'DA KULLANICI DUYGULARININ KOLLEKTİF SINIFLANDIRIMI

Ileri, Ibrahim
Yüksek Lisans, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi : Doç. Dr. Pınar Karagöz

Ağustos 2015 , 75 sayfa

Sosyal ağların son zamanlardaki hızlı yükselişi, çeşitli konular hakkında kullanıcı fikirlerini içeren büyük bir miktarda veri üretmiştir. Kullanıcı gönderilerinin görüşlerini sınıflandırmak adına, birçok metin tabanlı teknikler literatürde yerini almıştır. Görüş analizinin devamı olarak, duygu analizi üzerinde de çalışmalar bulunmaktadır. Çok sayıda farklı duyguların bu noktada ele alınması gerektiğinden dolayı, problem çok daha karmaşık bir hale gelmektedir. Bu tez çalışmasında, birbirlerine bağlı kullanıcıların benzer duygular barındırmalarına daha meyilli olması üzerine kullanıcı odaklı farklı bir yaklaşım kabul edilir; bundan dolayı, ilişki bilgisinden yararlanılması, sosyal ağlarda kullanıcı düzeyinde görüş çıkarım işini tamamlayabilmektedir. Deneysel bir veri kaynağı olarak Twitter ele alınarak ve önerilen kolektif sınıflandırma algoritmasıyla çalışılarak, kullanıcıların konu üzerinde bilinmeyen duyguları etkin ve işbirlikçi bir ortamda tahmin edilmektedir.

Anahtar Kelimeler: Sosyal Ağlar, Görüş Analizi, Kollektif Sınıflandırma

To my lovely family and friends who are always with me...

ACKNOWLEDGMENTS

I would like to thank my supervisor Associate Professor Pınar Karagöz for their constant support, guidance and friendship. It was a great honor to work with her and our cooperation influenced my academical and world view highly.

I would like to offer my great respect and love to all of the instructors of our department. Especially, I am sophisticated with their superior wisdom on their fields during my graduate lessons.

There are a lot of people that were with me in these days. They supported me, they made me who I am, they are true owners of this work. It is not possible to write down why each of them is important to me and this work, because it will take more space than the work itself.

I am very grateful to all people I know during my research assistantship in METU-CENG, they changed me deeply: my vision towards life, happiness and friendship. I am very lucky to have them all. So I'll just give names of some of them; Abdullah Doğan, Ahmet Rifaioğlu, Murat Öztürk, Alperen Eroğlu, Burak Kerim Akkuş.

Lastly, sincerest thanks to each of my family members for supporting and believing in me all the way through my academic life.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

World Wide Web (WWW) is a huge data warehouse with its wide range of applications. Users have great opportunities to introduce their opinions on products and discuss specific topics with their friends. Therefore, web sites such as Facebook, Twitter or Epinions are turned into attractive platforms not only for end-users but also for researchers. Obviously, this situation introduces new dimensions and research problems for the feature engineering field.

Recent advances in social networks increase the ways of explaining ideas on diverse subjects. However, users can share their opinions with their online friends in a collaborative manner. On the other hand, connected users may also write about quality of their favorite products to rank them. All that rich information sources make the social networks a suitable working base for researchers. New problems are waited to be solved and new applications are needed to be developed by them to help the corporations future orientations.

Effective methodologies and techniques are required to extract various kinds of information from social networks automatically. One of them which is identifying users' sentiments on a product or service has turned into a valid indicator of marketing success. This area could be seen as Sentiment Analysis on Social Networks. Apart from the classical sentiment analysis algorithms, networked data include valuable relationship information that can contribute to this analysis process beside texts that are produced by the users.

From the social networks view, users and their friendships are represented as nodes

and tied edges in order. By analyzing the interactions between users and finding the opinion orientation for partial of them, it is possible to infer the other remaining users opinions which are previously unknown. This process is named as networked data labeling problem and explained in Section 1.2.

1.1 Social Network Analysis and User Relationships

Online social networks are often called as social network sites where users can create their profile pages and send messages or share their opinions with their other connected social relationships. In other words, they are user generated content networks that allow users to express their sentiments and communicate each other.

Social network analysis includes theories, tools and methodologies for understanding the network structure and underlying relationships. Generally, social network researchers collect the networked data, analyze the data with some special tools and aim to find important patterns from related instances of the network. In terms of machine learning, patterns are learned by considering dependencies between entities in a supervised or unsupervised way. At this point, effective methodologies are needed to be designed unlike simple traditional learning algorithms.

Social networks are divided into three types by their connection characteristics [15] such as:

- Ego-Centric Networks: are those networks that are connected with a single node or individual. E.g. my good friends (on Twitter).
- Socio-Centric Networks: are networks in a "box". E.g. connections between children in a classroom.
- Open-System Networks: are networks in which the boundaries are not clear. E.g. the elite of the United States

However, from the point of sociological view, there are some social situations that invoke a relationship between one node to another. Each of them [15] could be defined as:

- Propinquity: At all levels of social analysis, entities are more likely to be connected with one another, other conditions being equal, if they are geographically near to one another. More broadly, it could be defined as being in the same place at the same time. Two people tend to become friends it they are geographically close.

- Individual-level Homophily: Homophily (From the Greek, “love of the same”), is the sociological term which is generally known as analogical statement: "birds of a feather flock together." At the individual level, people are more likely to be friends if they share common characteristics. In brief, there are four processes [15] to invoke friendship between people such that:

    1. The same kinds of people come together;
    2. People influence one another and in the process become alike;
    3. People can end up in the same place;
    4. Once they are in the same place, the very place influences them to become alike.

On the other hand, there are some useful social theories in micro-blogging to facilitate social network analysis. *Sentiment Consistency* suggests that the sentiments of two messages posted by the same user are more likely to be similar than those of two randomly selected messages. *Emotional Contagion* reveals that the sentiments of two messages posted by friends are more likely to be similar than those of two randomly selected messages. *Frequency of communication* reflects accurately on the emotional content, and amount of influence in a relationship.

A social network is formally represented by a graph, where nodes are users and edges are relationship links between them that can be either directed or not due to the different interpretations of relations. Users connect to each other in an unidirectional (e.g. following relationship on Twitter) or bidirectional (e.g. friendship relationship on Facebook) ways. Also, messages are the main features of the nodes. Users send messages to rank products, discuss political views, express their emotions etc. A message can be defined by a text which targets specific people.

1.2 Networked Data Labeling Problem

As mentioned in Section 1.1, social networks are represented as graphs. Finding/learning the class labels for some nodes, need special treatments. There are two major approaches for networked (linked) data classification as follows:

Local approaches (iterative classification) include these steps:

- Training a model locally using only labeled data
- Applying the learned model iteratively to classify unlabeled data

Global approaches (collective classification) include these steps:

- Using unlabeled or partially labeled data and link information for learning
- Approximately inferring the unknown labels by using specific feasible methodologies

More technically, networked data labeling problem could be seen on related Figure 1.1 and stated as follows:

Given: An undirected graph *(V, E)*, where E is the set of edges and each node in V corresponds to a vector of features A_1,..., A_n, C, where C denotes the class attribute; the values for the A_i are known for all $v \in V$, but class labels for C are only known for a proper subset T of V. A node is also called as a data instance.

Find: The class labels for $U := V - T$.

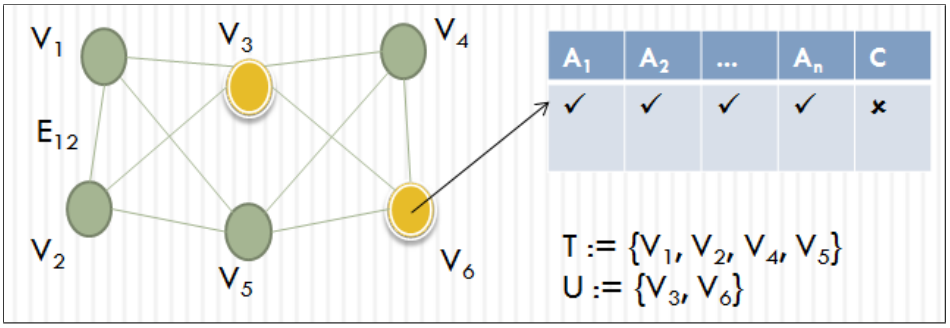

Figure 1.1: A Partially Labeled Sample Graph for Networked Data Labeling Problem

As seen on Figure 1.1, there is a sample partially labeled graph which contain six nodes and ten edges between them totally. It follows the above notations for the

definitions such as proper subset T and unknown labeled nodes set U. Two nodes (V_3 and V_6) do not contain class attributes for this graph. For the node V_6, its table of attribute list which is indicated by a near arrow (↗), shows the presence of attributes with cross (✗) and check signs (✓).

1.3 Contribution and Organization of Thesis

In this thesis, collective classification algorithms which constitute a sub-field of link mining field, are applied on the context of emotion analysis. Twitter users are nodes and their relationships are edges which are extracted from Turkish retweets or including user mentions (@) Turkish tweets. Giving graph structure as an input to collective classification framework, unknown emotion labels for users are predicted by utilizing their labeled neighbors relationship information.

Proposed relational classifiers are also experimented with different configurations. SVM is used as a baseline methodology for comparison purposes. Since Turkish tweets are already raw texts, some special text mining techniques are applied to turn them into meaningful entities by targeting feature vector construction. However, apart from this, all of the remaining methods are equally applicable to the texts in other languages as well. With the aim of applying collective classification techniques on the context of emotion analysis in social networks, to the best of our knowledge, this is the first work in the literature.

The contribution of the thesis can be summarized as follows:

- As the first work in the literature, collective classification algorithms are applied on the context of emotion analysis in social networks.
- Different Twitter datasets are gathered with their generated relationship information.
- Some language-specific text mining techniques are applied for Turkish tweets processing.
- SVM is used as a baseline methodology for the comparison purpose.

The thesis is organized as follows: Section 2 presents a survey of the related studies in literature which include classical text based and link-based sentiment analysis methodologies. Section 3 describes Twitter world, employed classification methods, feature vector construction and the tools that are used for processing and development. Section 4 presents mainly proposed methodologies for data gathering, preprocessing, relationship generations and collective classification task. Section 5 shows the related experiments and their results. Lastly, Section 6 concludes some useful remarks about this study and gives future directions.

CHAPTER 2

LITERATURE SURVEY

Categorizing sentiments gathered from online data sources has greatly increased in the academic world in recent years. All of these methods of the literature use the intractable text segments such as words, sentences, etc. for the task of inference. Some of these are mentioned in Section 2.1.

Emerging rise of micro-blogging on social networks, people have chance to express their emotions on such platforms easily. Analyzing user's tweets or classifying user's emotions on social networks are the subject of many studies in the literature. So, emotion analysis has been the distinct focus of researcher's work and some of these are summarized in Section 2.2.

In this thesis, the proposed approach aims to improve the accuracy of the model by employing relationship information between users. It also broadens the classification task's horizons due to the fact that it could be applied on users whose sentiments are not known before. Since, there is only one work [29] which uses link information in the context of sentiment analysis in literature, some similar link-based classification approaches are explained in Section 2.3.

2.1 Classical Sentiment Analysis

Classical sentiment classification could be thought as a text classification problem. Traditional text classification categorizes documents into different subjects by using subject-related words as the key features. However, in sentiment classification,

sentiment words such as good, bad, amazing, etc. that indicate positive or negative opinions are much more valuable.

By looking from the text classification's point of view, any supervised learning method can be applied on sentiment classification. As an example, [26] is the first study to categorize movie reviews into two binary classes as positive or negative. A bag of words (unigrams) approach is employed as features combined with supervised classification methods that are naive bayes or SVM. Also, other feature options are tried by the authors and has led great results.

In later works, many more features and learning methodologies have been studied by a large number of researchers. Similar to other supervised machine learning tasks, the key point for the sentiment classification is the mining of valuable feature sets. Some of them are exemplified as follows:

- Terms and their frequencies: Individual words and their related counts as features.

- Part of speech: For example, adjectives are specially treated as more important features due to its strong indication of opinions.

- Sentiment words and phrases: Good, wonderful, etc. are positive sentiment words and bad, poor, etc. are negative sentiment words. They are weighted differently for related classes.

After these feature identifications, classification step is mainly based on lexicons. By using valuable words and phrases, sentence opinion orientations are tried to be inferred. On the other hand, special treatments such as negation handling or oppositional clauses are considered occasionally. For example, in [10], sentimental words and phrases were identified and sentences were scored due to the previously constructed lexicon firstly. Then, negations are handled to tune the sentiment scores. Oppositional clauses are used to find the missing scores of some sentences which can not be determined on previous steps. Finally, individual scores are summed up and sentences are classified into their sentiments.

2.2 Emotion Analysis

Since the user's emotional states are subjective, they can reflect people's attitudes towards different domains including news, tweets, blog contents. Generally, there are two main approaches in the literature for emotion detection on texts. The first one is the text classification based methods which build classifiers from labeled text data as in traditional supervised learning. The second method detecting the emotional states especially in tweets is the lexicon-based approach. The emotion lexicon is created for this approach.

Regarding the field of psychology, Ekman [11] defined 7 emotions which are categorized by observable human facial expressions. Kozareva et al. [18] classified news headlines using these identified emotion classes. They averaged different web search engines hit counts on the emotional classes and news headlines as query words. In order to measure the co-occurrence of the query words of the headline and emotion class labels, they employed a method, which is called PMI (pointwise mutual information).

Aman and Szpakowicz [4] combined unigrams with emotion lexicons and a thesaurus as features to classify blog sentences into 6 emotion categories.

Alm et al. [3] presented empirical results of applying supervised machine learning techniques to categorize English fairy tale sentences into different emotions. The different emotion classes used in this study were angry, disgusted, fearful, happy, sad, positively surprised, negatively surprised. They proposed their own text-based classifier algorithm (SNoW) and it achieved higher accuracy results than Naive Bayes classifier or bag of words approach.

Go et al. [12] applied supervised learning methods to classify crawled Twitter data into binary sentiments as positive or negative. Following that, emoticons are treated as noises and removed from each tweet. Then, features are constructed with different n-grams and part of speech methodology. Lastly, Naive Bayes, Maximum Entropy and SVM classifiers are experimented. SVM classifier behaved best all in all. It achieved 80% accuracy when trained with emoticon data.

2.2.1 Emotion Analysis on Turkish Texts

Torunoğlu et al. [33] analyze pre-processing effect on the classification of Turkish texts. They crawled their data sets from Turkish newspapers. By applying pre-processing methods including stemming, stopword filtering and word weighting, their experiment results showed significant improvement on classification accuracies.

Boynukalın [5] worked on two data sets. One of them is the Turkish translation of ISEAR [1] data and the other is the manually labeled Turkish fairy tales. Emotion levels are predicted using different n-gram feature constructions and weighted log likelihood algorithm [24] is utilized to determine most significant features.

Akba et al. [1] investigated the feature selection methods on Turkish movie reviews. They selected these reviews due to their huge amount of possible emotional information. They labeled their corpus by dividing emotions into three categories as positive, negative and neutral. In their experiments, support vector machine (SVM) and Naive bayes had been employed for classification and F_1 score was used for performance evaluation. Different SVM performance results were obtained depending on the classification of movie reviews into two or three categories. Binary classification achieved the best result as 83.9%.

Toçoğlu et al. [32] proposed an emotion extraction system from Turkish texts which is based on text classification approach. They gathered data set due to survey answers of 500 university students. In this questionnaire, students are asked to describe their most intense moments for 7 emotion categories which are happy, shame, guiltiness, disgust, sadness, angry and fear. Then, texts are pre-processed and modeled with Vector Space Model with tf-idf weighting scheme. Finally, applied Naive Bayes classifier in Weka achieved around 86% promising accuracy result.

Demirci [9] classified Turkish tweets into six emotion categories (anger, surprised, fear, sadness, joy and disgust) with supervised learning. Raw tweets are collected according to emotional hash tag lists and automatically labeled. Then, some preprocessing are done to get clean texts. Also, beside the special treatments mentioned above, some morphological analysis are performed because of the sarcastic nature

[1] http://www.affective-sciences.org/researchmaterial

of Turkish language. Features are constructed based on vectors with different combinations of n-grams. Large feature vectors are minimized using feature selection methods. Finally, some supervised methods such as naive bayes, k-nn and SVM are compared with the baseline algorithm of Boynukalın [5]. SVM achieved approximately 69% accuracy result at best.

2.3 Link-Based Classification

Relationships between objects are common in socially networked data on the web. Those links represent valuable information on the objects explicitly or implicitly. For example, user's followers list on Twitter is considered as explicit relationships. On the other hand, statements such as "if many followers of a user hold similar sentiments on a subject, the user may hold the same sentiment too.", expresses an implicit relationship.

Generally, data mining algorithms try to find ideal patterns from each of the independent instances. When all of these instances are modeled as a graph, the task turns into learning only the node attributes except from relations between them. Nevertheless, leveraging relationship information contributes the quality of building models or algorithms positively. World's most used web search engine Google has taken its roots behind their designed PageRank [6] algorithm. However, many link mining approaches need to deal with heterogeneous data which include different types of instances and relations.

Since link mining leads on a wide scope, a subfield of it, called collective classification, is used in this study. Briefly, it aims to predict the labels of objects with the relationships among them. The main challenge is to design an algorithm for collective classification that uses associations between object classes and jointly infer their labels in the graph.

Chakrabarti et al. [7] consider categorizing related news objects in the Reuters dataset. They are the first to leverage class labels of related instances and also their attributes. Nevertheless using class labels improves classification accuracy, the same thing does not apply for considering attributes. Also, they use previously proposed relaxation

labeling inference algorithm [30].

Neville and Jensen [23] propose simple link based classification methodology which classifies corporate datasets that represent heterogeneous graphs with many different set of features.

Lu and Getoor [19] aims to enhance traditional machine learning algorithm by introducing new features which are built out of correlations between objects. As a result, a new link based classification algorithm which uses probability terms such as Markov blanket of related class labels of classifying objects, has come into existence.

Pang and Lee [25] seeks to determine sentiment polarities of movie reviews by extracting subjective portions of the sentences. For this purpose, they use a graph-based technique that finds the minimum cuts. It deals with psychical proximity between related sentences by using the idea that "texts which are near each other may share the same subjectivity labels". They formulate such correlations with calculated scores and estimate objective functions with Naive Bayes and SVM methods. By this way, contextual information is added in polarity classification process and lead to significant improvement for accuracy.

Shivashankar and Ravindran [31] analyze multi grain sentiments on review articles that are gathered from websites like CNET, Epinions and Edmunds. They propose a multi grain collective classification algorithm for the partially labeled data at different levels. Document, sentence and tuple levels are represented as graphs with their inter or intra relations between them. Unknown sentiments of documents are inferred from partially labeled nodes by employing iterative collective classification algorithm. It is already a binary sentiments. As a baseline, lexicon based classifiers for two different levels are evaluated. Proposed algorithm improves the accuracy results around 30% in terms of information retrieval metrics.

There is not much work in the literature with exactly the same purpose in this study. There is only one work that is Juliano et al. [29] proposed a user centric approach on the context of sentiment analysis. They have classified Twitter user's political opinions into two binary classes by using collective classification. Their algorithm takes a partially labeled graph, applies a graph pruning process and runs the collec-

tive classification. Preliminary experiments on data crawled from Twitter have shown promising results. Their results show that, when the labels of 10% of the most connected nodes are known, their model is able to classify the remaining unknown nodes with almost 80% of accuracy.

CHAPTER 3

BACKGROUND

3.1 Twitter

Twitter is a popular social microblogging service with millions of registered users who share their sentiments on any topic. Twitter users post short text messages, or *tweets* which are limited up to 140 characters long. Users follow others or they are to be followed. In terms of social connections, following relationships do not need to be bidirectional. In other words, the user who is being followed, not necessarily follow back. Followers can see all the messages from those the user follows.

There is a special mark-up vocabulary for answering the tweets. *RT* stands for retweet, @ is followed by a user identifier indicates the user, and # is followed by a word represents a hashtag. By the retweet mechanism, users can have facilities to spread their ideas in a quick and efficient manner.

Twitter users can reference other user via using the mentioning/replying mechanism that includes @*username* phrase in tweets. In this way, a link is created between the message in the users profile page. Mentioned tweets are seen on the referenced user's account and this results detecting the tweets mentioning them.

Also, due to privacy reasons, some Twitter users can restrict their posted messages to only their followers. In this study, publicly available user's profiles and (re)tweets are crawled. Besides, the posted messages are already available to anyone by default.

3.1.1 Emotions and Networks on Twitter

Social sharing of emotions is one of the main purposes for using blogging services such as Twitter. Twitter users can interact with other users by using following ways:

- Forwarding (retweet) messages from other users
- Reply to or mention other users in their messages

In this way, it is expected to find a correlation between emotion contagion and network's properties. In [17], due to different types of interactions, two research questions are investigated statistically such that:

1. What is the correlation between user's inclination towards sharing emotions in their posts (retweets or mentions) and their number of followers?
2. What is the correlation between user's inclination towards sharing emotions in their posts (retweets or mentions) and their network characteristics such as density?

Although there are not strictly definitive results, it is showed that expression of emotions are associated with more followers (friends) and sparser (highly cluster-able) networks on Twitter. In other words, there is not an exact rule such as sharing emotions need much followers or vice versa. Possibly, because of the social constraints, users may hesitate to share emotional experiences with dense contacts. Also, interaction networks are needed to be more concentrated for future works.

3.2 Classification Methods

3.2.1 Support Vector Machines

Support Vector Machines (SVM) [14] aim to find some boundary points, called as *support vectors* from each class in feature space and construct a linear function which separates them into wide area. By including extra non-linear terms, (*kernel*) functions

cover high-dimensional data and form quadratic, cubic, and higher-order decision boundaries (*hyperplanes*).

In other words, SVM (methods) map the data into a new feature space that contains nonlinear *hyperplanes* via supervised learning of linear models. It is focused to find the *maximum-margin hyperplane* (or linear model) which has the closest distance to each *support vectors*.

For high-dimensional spaces following finding *maximum-margin* hyperlane Equation 3.1's parameters are needed to be learned by SVM:

$$max\_margin\_hyperplane = \beta + \sum_{i \in S} \alpha_i \times c_i \times (s(i) \cdot s)^n, \tag{3.1}$$

where S is the set of support vectors, c(i) is the class value of ith support vector, *n* is the dimension (factor) count, α and β are the kernel parameters, (s(i) $\cdot$ s)^n is the dot product of ith support vector (training instance) and the test instance. Such dot product of these vectors corresponds to *polynomial kernel* function. As a special case, if *n* = 1, it is called as *linear kernel* function.

According to the computational complexity problem, if the mapped space is a high-dimensional one, *kernel* functions, that are based on the dot product of feature vectors, are needed to calculated before mapping is done. If the number of features is too large, there is no need to map data into a higher dimensional space. *linear kernel* function serves the purpose sufficiently.

For optimal learning, different value combinations of kernel parameters (α and β in Equation 3.1) should be tried with grid search and k-fold cross validation techniques.

In k-fold cross-validation, training data is divided into into k parts of equal size. Then, one part is tested using the classifier trained on the remaining k-1 parts sequentially. Finally, each instance of the whole training data is predicted once so final accuracy is the percentage of data which are correctly classified. Grid search tries different combinations of parameters using cross-validation.

3.2.2 Collective Classification

Networked data has many different scenarios. It is an important issue to investigate how objects influence each other in network. For example, how user's emotions are affected by his/her relationships in Twitter. On the other hand, product's attributes are tried to be improved by examining the ranking results of different networked users. All of these intersect at one common point which is finding the labels of all entities in the network.

There could be three different types of relations in a networked data which can be used to label network objects:

1. Relations only between object's own labels and (local) attributes
2. Relations between object's own labels and its adjacent neighbors known attributes (also, known labels)
3. Relations between object's own labels and its adjacent neighbors unknown labels (also, known attributes according to the approach)

Collective classification utilizes all kind of relations that are listed above to find the labels of all network objects. By the way, relational classification field uses only the first and second. Main difference between two fields is that in partially labeled networks, unknown labels of all objects are needed to be simultaneously inferred. This can only be achieved by using collective classification techniques.

Working principle of the collective classification is shown on Figure 3.1. Given networked data, local classifier produces prior class membership estimations for unknown labeled (test) nodes. Relational classifier is the core component which includes different options such as summarizing test node's neighborhood information or integrating with external Weka classifiers. Consequently, inference method is fed with local and relational classifier's trained models that are used for simultaneously estimating class label values for test node instances.

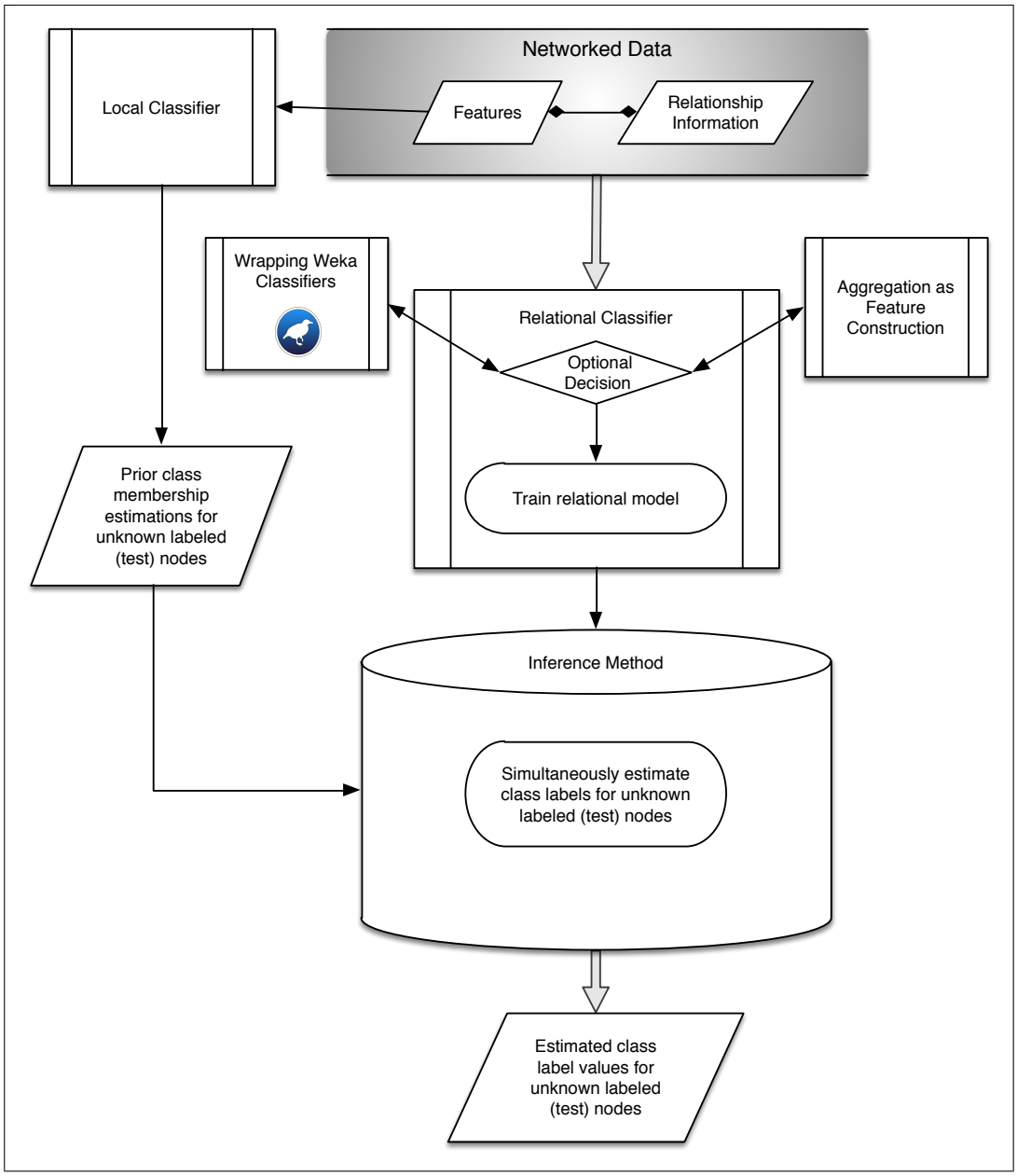

Figure 3.1: Working Principle of the Collective Classification

3.2.2.1 Univariate Collective Inferencing

Here, univariate collective inferencing or within-network classification, a special case of collective classification, is described. Univariate means that inferencing task is applied by considering only the class labels of neighboring nodes, not using local attributes of them. More precisely, marginal probabilities of class membership of a particular node are calculated based on the class memberships of other connected

nodes in the network.

Formally, univariate collective inferencing could be defined with the similar notations seen in Chapter 1, Section 1.2 as follows:

Given: Graph G = (V, E, C) where C_i is the (single) attribute of node $v_i \in V$, and given known values c_i of C_i for some proper subset of nodes T,

Task (Univariate collective inferencing): Simultaneously inferring the labels c_i of C_i for the remaining vertices, $U := V - T$, or a probability distribution over those labels.

While doing collective inferencing task, some assumptions are needed for the probability calculations. First of these stands for the relational learning and a first-order Markov assumption as given in Equation 3.2:

$$P(c_i|G) = P(c_i|N_i), \tag{3.2}$$

where N_i is the immediate neighbors of node v_i. Local neighborhood is considered to be independent from all other remaining nodes in the network ($V - N_i$) by the relational learning algorithms.

Secondly, it is assumed that learning a part of the whole network (a particuar node and its neighborhood), reflect the probability dependencies which are also valid for the general.

3.2.2.2 Aggregation as Feature Construction

Collective classification can use constructed relational features for the inference task. However, most of the local classifiers use only fixed-size feature vectors, neighbor counts are varied greatly in networked data. For instance, a Twitter user can have many followers. Although it is not a preferred method, by considering limited (equal) number of connections for each user, fixed-size feature vectors could be constructed.

A desirable solution is to apply aggregation techniques to summarize the node's neighborhood information. For example, the number of neighbors which have different class labels could be counted and added as a new feature to node. Class labels

may be replaced or supported with local attributes. For numerical attributes, it is also possible to use statistical methods such as minimum, maximum, median, mode, ratio.

On the other hand, for each pair of neighboring nodes, similarities of their local attributes can be considered exactly. In this study, a similar method is discussed but not only implemented as an aggregation method but also used as a weight in relational probability calculations. Perlich and Provost [27] have worked on aggregation-based feature construction as the relational concept in detail.

3.2.2.3 Collective Classification Algorithms

Collective classification can be seen as a relational optimization task for networked data. According to algorithm's nature, different relational objective functions are optimized within collective inference techniques.

It is better to divide collective classification into three models for two reasons [20]. Firstly, differences among them are seen clearly. Secondly, different models can be combined or mixed to propose alternative systems.

These models are described as follows:

1. **Local (non-relational) model:** This model is learned for target (class) variable by using the local attributes of the nodes in the network. Alternatively, classical machine learning methods can be employed. Generated initial class priors stand for other two components subsequent use.

2. **Relational model:** Relational features and links among entities come into prominence for this component. It builds different objective functions which are ready for estimate node's target attribute probabilities with its neighborhood. It is also possible to benefit from local attributes of the neighboring nodes.

3. **Collective inference:** Created relational objective functions are generally the joint probability distributions which are based on Markov Random Fields. For example computing Equation 3.2 needs collective inference methods. As a

result, it is tried to be learned how a node's classification is influenced from its neighbors classification in a collaborative setting.

In the next subsections, according to above components, local, relational classifiers and collective inference algorithms (which are also implemented in Netkit-SRL, Section 3.4.3), are introduced appropriately.

3.2.2.3.1 Local Classifiers

Concisely, local classifiers generate initial prior class membership estimations from the networked data. These priors can be Bayesian or directly computed from local attributes of the node instances independently.

Formally, given nodes with known class labels as training data, local classifiers learns a model that represents given node's probability distribution for class memberships. Each node is treated as independent from other nodes (like in traditional machine learning) and local attributes are considered.

3.2.2.3.2 Relational Classifiers

Similar to the link-based classification field (Chapter 2, Section 2.3), relational classifiers learn a relational model via producing joint probability distribution over the nodes neighborhoods on given networked data.

On the other hand, aggregation techniques could be utilized to help to summarize neighborhood information for each node. By this way, neighbor's features gain more importance for learned model. E.g. for a particular training node, local attributes and class labels of neighbor nodes could be combined with naive bayes model [7].

Here are the relational classifiers that are already implemented in Netkit-SRL and used in this thesis experiments:

Weighted-Vote Relational Neighbour Classifier (wvrn): Weighted-vote relational neighbour classifier produces a weighted mean of class membership probability estimations from node's neighbors. Also, homophily principle (Chapter 1,

Section 1.1) are assumed. Instead of training a separate relational model, it estimates the probabilities at the inference time simultaneously.

By following the notation in Chapter 1, Section 1.2, objective function is defined as:

$$P(c_i = c|N_i) = \frac{1}{Z} \sum_{v_j \in N_i} weight_{i,j} \times P(c_j = c|N_j), \tag{3.3}$$

where Z is standard normalization variable which smooth the summed values on the range 0 and 1. $weight_{i,j}$ represents the edge weight between node *i* and node *j* (simply equals to 1 for this study).

Probabilistic Relational Neighbour Classifier (prn): This is a special case classifier for the weighted-vote relational neighbour classifier and uses naive Bayesian combination of neighbors edges. Simply, while estimating a particular node's class label probability, it multiples each neighboring node's class prior probability values.

Class-Distribution Relational Neighbour Classifier (cdrn-norm-cos): The class distribution relational neighbor classifier creates an average *class vector* for each class of node and then estimates a label for a new node by calculating how near that new node is to each of these *class reference vectors*.

Class vectors are calculated by the summation of weights that are linked to each neighbor's (known) class. E.g. if there are six neighbors and two classes in networked data, there are twelve elements in this node's class vectors. Also, average is obtained by dividing the vector elements with the number of times that class was found.

Class reference vectors are the average of the class vectors for nodes known to be of class *c* based on normalized vector summation.

Finally, ready for estimation objective function is defined as:

$$P(c_i = c|N_i) = similarity(classvector(v_i), refvector(c)), \tag{3.4}$$

where cosine similarity with standard normalization is used by default in Netkit-SRL.

Network-Only Bayes Relational Classifier (no-bayes): Network-only Bayes classifier is based on the algorithm from Chakrabarti et al. [7]. It uses multinomial naive Bayesian objective function which includes v_i's neighbors as:

$$P(c_i = c|N_i) = \frac{P(N_i|c) \times P(c)}{P(N_i)}, \tag{3.5}$$

and

$$P(N_i|c) = \frac{1}{Z} \prod_{v_j \in N_i} P(c_j = c_j^{'}|c_i = c)^{weight_{i,j}}, \tag{3.6}$$

where Z is standard normalization variable and $c_j^{'}$ is the class label observed at node *j*. Denominator of Equation 3.5 is ignored due to normalization across the classes.

Actually, in Netkit-SRL, there are some differences from the original algorithm. For example, Chakrabarti et al. used local attributes of neigbor nodes. However, Netkit-SRL approaches the algorithm in univariate environment and does not use local attributes for initial priors.

Due to the collective inference algorithm choice, class priors or null values are employed for neighbor nodes. For possible zeros in probability estimations, Laplace smoothing is applied. Also, it is assumed that all neighbor nodes have known (or estimated) class labels for future steps of inferencing.

Briefly, Network-only Bayes classifier counts the class labels of node v_i's each neighbors. Then, product it with prior class distributions. Estimation needs product of each neighbors observed class value probabilities conditioned on given nodes class values and getting powered with edges weights.

Network-Only Link-Based Relational Classifier (nolb-lr-distrib): Network-only link-based classification is based on the algorithm from Lu et al. [19]. Firstly, it creates normalized feature vector of the training node via aggregating its neighbor's attributes (only class attribute for Netkit-SRL). Then, it uses logistic regression (wrapped from Weka for Netkit-SRL) for relational modeling. Learned model is employed for the estimation of P(c_i = c|N_i) in collective inferencing step.

Even though there are several aggregation methods in Netkit-SRL, feature vectors consist normalized count of neighbor class attributes by ratio aggregator.

Wrapped Relational Classifiers from Weka: Netkit-SRL has proposed a wrapper class for using Weka's local classifiers as relational classifers. These are logistic regression, multinomial naive bayes (naivebayes) and decision trees (j48).

By using aggregation techniques, networked data is turned into Weka's desired classification features and then related algorithms is used for predictions. With this technique, ideal attributes for each node instance gain more importance while learning relational model and doing inference in collective classification step. The more useful local features for a node, classification results are the better. In this study, it is supported by the experiments results (Section 5.3.2) which show best accuracies for the thesis emotion data.

Decision tree (j48) is the implementation of Iterative Dichotomiser-3 algorithm (ID3) developed by the Weka [28]. It is based on the information gain idea that iteratively builds a tree (model) according to determined splitting criteria as the attribute which has highest information gain score. Then, classifies unknown labeled object by adding it as a leaf into the true place on tree. That true place is found by following splitting criteria node values compared with local attribute values of processed node.

3.2.2.3.3 Inference Algorithms

The main goal of collective inference is to infer the unknown class labels of nodes by maximizing the marginal probability distribution which is represented by learned objective functions from relational classifiers. Also, if there is need for estimation of interested node's class membership, they return a prior estimations from local classifiers.

In this section, two collective inference methods that are already implemented in Netkit-SRL and used in this thesis experiments, are described as following:

Iterative Classification: Concisely, iterative classification classifies the node's unknown class labels by updating current state of the graph in each iteration until every node's label is stabilized or maximum iteration count is reached. In other words, it applies a local classification to every node, conditioned on the cur-

rent probability estimations of its neighbors and iterates until local objective function estimations converge to a stable solution.

It is based on the methodology of Lu et al. [19]. Pseudo-code for Netkit-SRL implemented iterative classification is seen on Algorithm 1.

Algorithm 1 Pseudo-code of (Netkit-SRL) Iterative Classification

for all $v_i \in U$ **do**

$init\_prob\_vec \leftarrow$ local classifier estimations which are *null* values for each neighbor node's labels

end for

Randomize each $v_i \in U$ as R

repeat

for all $v_i \in R$ **do**

$init\_prob\_vec \leftarrow$ relational classifier estimations which are *non-null* values for each neighbor node's labels

Assign the class value which has the maximum probability value in $init\_prob\_vec$ to v_i

end for

until Iteration count is reached to 1000

Relaxation Labeling: Relaxation labeling is based on the method of Chakrabarti et al. [7]. Pseudo-code for Netkit-SRL implemented relaxation labeling is shown on Algorithm 2.

Unlike iterative classification, relaxation labeling uses direct class estimations from learned models rather than constant labeling (e.g. as *null*). By this way, it does not miss the previously estimated probabilities in each step of the inference. However, this forces the chosen relational classifier to get all (unknown labeled) node's ($v_i \in U$) probability class distributions initially.

As seen from Algorithm 2, relaxation labeling does not follow the routine that estimates a single node at a time and updates the graph state instantly. It stands, holds the estimations from previous iteration then use these values on the next iteration. As a result, inference is carried out simultaneously.

By the way, Netkit-SRL uses simulated annealing as taking precautions for

Algorithm 2 Pseudo-code of (Netkit-SRL) Relaxation Labeling

for all $v_i \in U$ **do**

$init\_prob\_vec \leftarrow$ local classifier estimations which are *unconditional marginal class distribution* via using known labeled nodes

end for

repeat

for all $v_i \in U$ **do**

$init\_prob\_vec^{itercount+1} \leftarrow$ relational classifier estimations for P(c_i = c|N_i) from previous $itercount$

Assign the class value which has the maximum probability value in last $itercount$ of $init\_prob\_vec$ to v_i

end for

until Iteration count is reached to 99

algorithm's not converging possibilities. By tuning some parameters, node's self estimations become more important rather than its adjacent neighbors.

3.3 Feature Vector Construction and Feature Selection

Features correspond to the raw texts (*tweets*) for gathered data in this study. Text-based data mining is a challenging process due to its unstructured and compelling nature. Some special techniques are needed to be applied for extracting meaningful entities from texts.

Representing text objects in a common pot, vector-based approaches are preferable. Each text feature is characterized by their words. Then, all of these words are hold in a pool for latter use (*bag of words*). For each instance, presence of the words from that dictionary pool are weighted with their occurrence counts. As a result, texts are turned into common feature vectors which include their word counts.

Of course some words are not useful for the classification task. Such words are called as *stop-words* and should be eliminated before feature selection. Remaining words can construct a vast amount of feature space and need a mechanism to reduce this redundancy.

3.3.1 Word UniGrams

N-grams of texts are widely used in text mining or natural language processing (NLP) applications. They can be defined as ordered set of words list that form the text. Main difference from *bag of words* approach is to keep ordering of words in each text feature. Unigrams are words, bigrams are the two word phrases, trigrams are three word phrases and so on.

However, big n numbers are not leaded better results in text mining literature. Also in study [9], unigrams are leaded best accuracy result for the SVM classification method and so that it is chosen as baseline method for this study.

If $n = 1$, it is called as unigrams and as an example, sentence "Tea plant grows in wetlands." is divided its five unigrams as follows: {Tea}, {plant}, {grows}, {in}, {wetlands}.

At next step, features (unigrams) are weighted with their occurrence counts in the sentence and builds a common vector form for all instances. However for our above example, since each word occur only at once, its feature vector is simply represented as: <...1, 1, 1, 1, 1...>.

3.3.2 Information Gain for Feature Selection

Information gain [16] is a feature selection mechanism that is used for finding the useful features which have significant effect on classification process. Other features are seen as redundant and must be eliminated to reduce dimensional feature space.

The goal of feature subset selection is to find a minimum set of features such that the resulting probability distribution of the data classes is as close as possible to the original distribution obtained using all attributes.

It is based on the idea of measuring the amount of information about class predictions, namely only available information is the presence of feature and its class distribution. The features which have great information gain score, are seen as significant. In other words, these scores say how much part of the information available in that feature

contribute towards classification or not.

3.4 Tools

In this section, tools that are used in this thesis are described.

3.4.1 Zemberek

Zemberek [2] is an open source natural language processing (NLP) framework for agglutinative Turkic languages. It is coded in Java and has special libraries to process the words in Turkic languages.

Zemberek allows the following operations on Turkic words such as spell checking, stemming, misspelled word correction, word suggestion. With its pre-defined language-specific alphabets, letters, roots and suffixes files, Zemberek morphological parser analyze input words.

Spell checking is available only for one letter correction. When it is queried for a word, it returns a list of possible suggestions of the queried word. First item is used as best result.

A sample query for a misspelled word ("mrhabalar") can print corrected suggestions with the following Java code via Zemberek:

```
Zemberek zem = new Zemberek(new TurkiyeTurkcesi());
String w = "mrhabalar";
for (String str: zem.oner(w))
{
  System.out.println(str);
}
```

3.4.2 LibSVM and InfoGainAttributeEval on WEKA

Weka [13] is a popular data mining tool which has many different traditional machine learning and attribute processing methodologies.

LibSVM [8] is a integrated wrapper for support vector classification on Weka. Its *.jar* file must be included in the class path of Weka library for the Java virtual machine. It provides several types of support vector machines for multi-class classification, regression, and one-class problems, and gives a choice of different kernel functions such as linear and polynomial (explained in Section 3.2.1). Cross-validation is employed for the parameter selection.

InfoGainAttributeEval is a Weka's attibute selection mechanism which evaluates attributes by measuring their information gain with respect to the class. It uses the *Ranker* search method and ranks the attributes due to their information gains. Also, it discretizes numeric attributes via using the MDL-based discretization method or binarizes them. Required top-ranked *n*-attributes can be gathered by changing the optional parameter.

3.4.3 NETKIT-SRL

Netkit-SRL (or Netkit), is an open source Network Learning Toolkit for Statistical Relational Learning. It is coded in Java and integrates with the Weka [13] data mining with machine learning tool. It allows to combine different type of components for relational classification on networked data. It instantiates five modules that are also mentioned in Section 3.2.2.3 as:

1. **Input Module:** Reads the given data into memory as a graph structure.

2. **Local Classifier (LC) Module:** Returns local classifier's prior class estimations for given unknown labeled node by using only its own attributes.

3. **Relational Classifier (RC) Module:** Returns relational classifier's class estimations for given unknown labeled node by using its own and also it's neighbor's attributes.

4. **Collective Inference (CI) Module:** Returns class estimations for given unknown labeled node by applying collective inferencing that uses relational and local classifiers bringing together.

5. **Weka Wrapper Module:** This module wraps the different Weka machine learning algorithms (e.g. multinomial naive bayes, decision trees, logistic regression) by turning given networked data instances into Weka recognizable entities. As a result, Weka's algorithms can be used either as a relational classifier or a local classifier.

All of above Netkit modules can be seen at top-level as different interfaces which are needed to be implemented for the different type of classifiers. They are described in Section 3.2.2.3 in detail and used in this study. Also, Netkit allows to design new classifier components and use them with different configurations. Core working principle of Netkit framework follows the steps on Table 3.1.

Table 3.1: Core working principle of Netkit framework

Given: Graph G with T (known labeled node set), $v_i \in U$, kind of LC, RC and CI
Train local classifier LC's model via G with T Train relational classifier RC's model via G with T Apply collective inference CI on $v_i \in U$ via RC's learned model
Return:: Estimated class label value for $v_i \in U$

Relational classifier (RC) is the main component of collective classification procedure. Specifically, since RC requires aggregation (Section 3.2.2.2) on neighbor nodes, Netkit includes four options for the aggregation as:

- *None*: Do not use any attributes for aggregation (classical machine learning)
- *ClassOnly*: Use only class attribute for aggregation
- *ExcludeClass*: Use attributes except from class attributes for aggregation (classical relational learning)
- *All*: Use all attributes for aggregation

However, not all of above aggregation techniques are implemented for Netkit relational classifiers. This case is open to development and the point is caught in this thesis. And also seven aggregation methods are also included in Netkit as: *count, exist, max, mean, min, mode and ratio*.

A sample Netkit collective classification configuration can be run from terminal with the following command:

```
java -jar ../lib/NetKit.jar dataset.csv -inferenceMethod iterative
    -rclassifier nobayes -aggregation ClassOnly
```

CHAPTER 4

PROPOSED METHOD

4.1 Overview

This chapter presents the methodology proposed in this thesis work. Main objective is to collectively classify users' unknown emotions reflected in the social network postings with the help of their relationship information. On the other hand, by the proposed relational classifier, their text features are also taken into consideration.

The first step is data gathering. Within this study, we used two data sets, where one of them is publicly available and the other one is from the literature. Beside these data sets, we gathered a new set of data from Twitter. After collection of data, since mining Turkish texts is a challenging issue, retweets are turned into effective feature vectors. In addition, aiming to create social network, synthetic and realistic relationship data are generated respectively. Finally, SVM and collective classification algorithms are applied to each dataset.

Mainly, there are five phases:

1. Data Gathering
2. Data Preprocessing
3. Feature Vector Construction and Feature Selection
4. Relationships Generation
5. Classification

Detailed overview of the proposed method is shown in Figure 4.1.

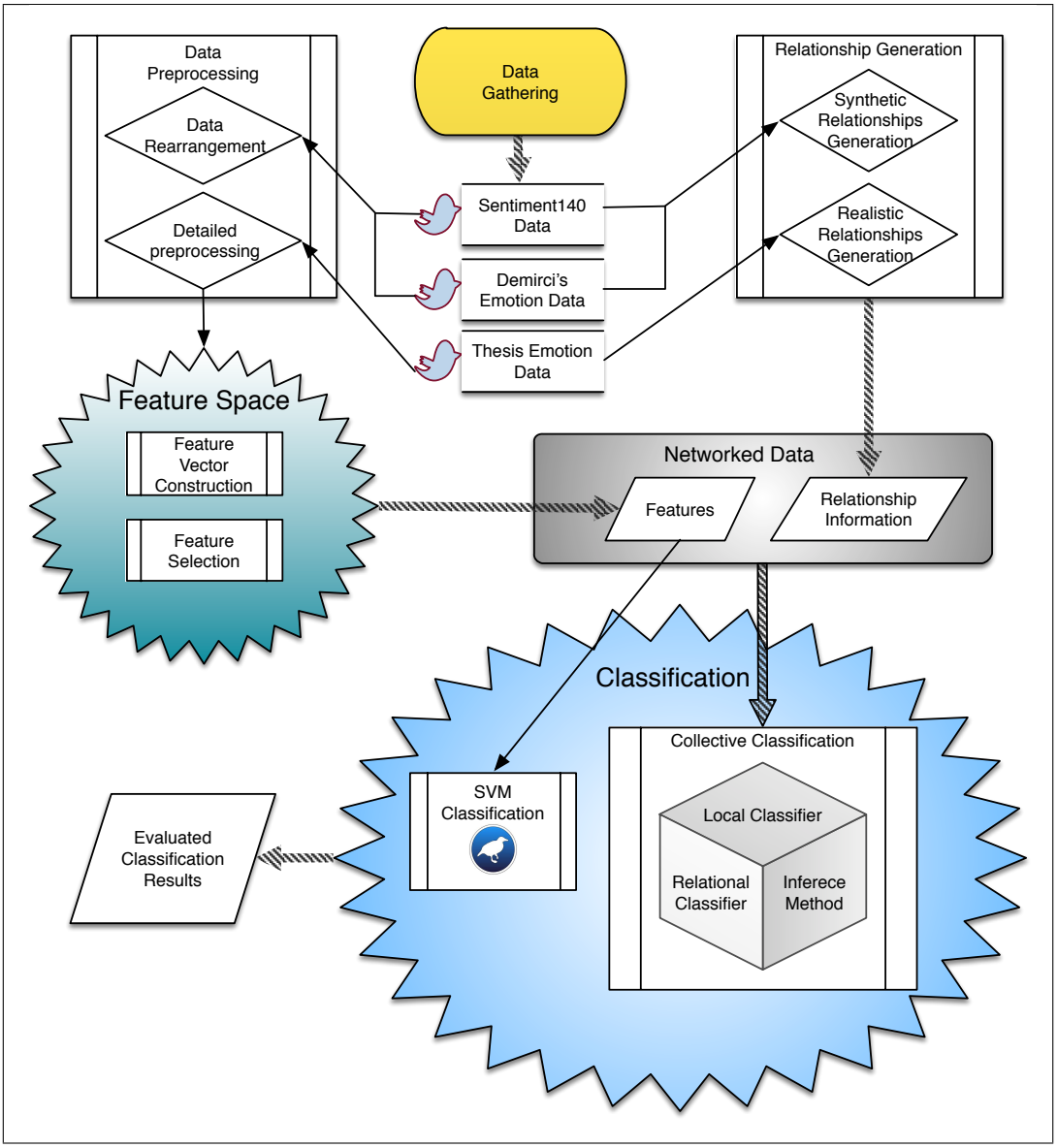

Figure 4.1: Overview of the Thesis Proposed Method

4.2 Data Gathering

Sentiment140 dataset is collected from Twitter API [1] and described in study [12]. They approach data collection issue as querying tweets due to constructed emoticon lists. For instance, while searching ":)" returns positive tweets, querying ":(" returns negative tweets. Final training data has equal class-distribution which con-

[1] https://dev.twitter.com/overview/api

tains 800.000 positive and 800.000 negative tweets. They also collect test data that consists of 177 negative and 182 positive manually marked tweets. However, since cross-validation evaluation method is used in this thesis work, test data is stayed out of need for the experiments.

Demirci [9] collects its data by querying each element of expanded emotion-related hashtag lists. For six emotions, there are six emotional hashtag words list is constructed with word's synonyms and adjective forms. Totally obtained 6000 Turkish tweets are equally distributed according to their class labels.

Similar to the work in [22], for gathering thesis emotion data labeled instances, different emotion related hashtags are monitored by using Twitter API. These are Turkish keywords such that "sinir" ("anger"), "korku" ("fear"), "mutluluk" ("happiness"), "üzüntü" ("sadness"), "iğrenç" ("disgusting"), "şaşkınlık" ("surprise"). If the word does not return enough results, its derived versions are employed. API restrictions are overcome by waiting for time intervals. As a result, 1200 labeled retweets are collected for each emotion category. Then, all of six categories retweets are merged together and there are 7200 instances in total.

4.3 Data Preprocessing

For the upcoming related Netkit experiments, Sentiment140 data preprocessing steps are:

- Unnecessary first, second and third fields are removed.
- Whole dataset is divided into two parts and each of them contains 400.000 positive, 400.000 negative instances. In this study, one of them are employed (800.000 instances)
- Since the study is a user-centric approach, duplicate user names are eliminated. Also, to be valid for all of the datasets that are used in this study, duplicate elimination follows a simple majority vote counting. With this method, there remains user instances whose class label has the largest majority.

- Along with unique users updated dataset contains 659.773 instances and all of them are shuffled.

After these steps, dataset is named as SentDS and final short view of the SentDS can be seen in Table 4.1.

Table 4.1: Short View of SentDS

Username	Tweet	Sentiment
alyssaavant	@ExtraordMommy my internet is down	0
itsnix9	@so_we_beat_on it makes me the winner and you the loser	4
Andi_Skene	All by myself... I like to be ALL BY MYSELF	4
neim81094	Dammit! Im forced to study! Why cant i just go 2 bed?!	0

Demirci's emotion dataset is also preprocessed in its original so that the hashtags that are used for classification purpose are stripped off from the tweets. It is named as EmoDS-1 and sample data instances are shown on Table 4.2.

Table 4.2: Short View of EmoDS-1

User ID	Tweet	Emotion
30	Sapsal olmayan bileti satıyor allahtan sağduyulu tiyatro personeli var da oturacak yer bulabildik..	anger
906	Topuklu ayakkabi giyip ayaklarini suruyerek dolasanlardan !	disgust
1625	Korkularının üzerine gidecen hacıı	fear
2553	Azicik gulelim ya..	joy
3242	sevgilim yok diyen hayatına bakıp şükür etsin tweeti atarken kullandığı ellerinden biri olmasaydı asıl mutsuzluğu o zman tadacaktı	sadness
4074	Patrona sorarak belirliyorlarmis bu asgari ucreti. dehsete kapildim	surprise

Besides all this, thesis collected emotion dataset's text features are needed to have detailed pre-processing for the latter feature vector construction steps. Such techniques are employed for turning each instance's raw text attributes (*retweets*) into noise-pruned forms.

Applied steps are shown in order as following:

1. Firstly, these are the noises that are cleaned from each retweet instance respectively:

   - New line, tab indicators
   - URL links, (@) usernames, RT flags

2. Then, emotional expressions, hashtags (inside tweets) and punctuation are extracted (hold for later use, if necessary) separately and removed. Especially, hashtags inside tweets are removed for preventing possible bias in future classifications.

3. Finally, all non-word characters are removed for other possible noises.

4.4 Feature Vector Construction and Feature Selection

After preprocessing steps of thesis emotion data, text-based attributes are represented as constructed feature vectors. Therefore, text-based mining techniques, which are described in Section 3.3, are applied.

For this purpose, each text feature is characterized by its extracted unigrams. Texts are divided into their tokens which are the root form of the words with Zemberek (Section 3.4.1) tool (also known as *stemming*).

Each individual token is also inspected for language and spell checking. Tokens that are not in Turkish are removed. If the word is misspelled, first corrected suggestion of Zemberek is founded for this word and returned this new suggested word's root as the stem.

All reliable tokens are added into a common pool (*bag of words*). The ones that are not seen as effective for the classification (*stop-words*) are eliminated from this pool. Elimination of these stop-words are done according to a Turkish stopwords list [2], retrieved from a free-to-use project which has collection of stop words in 29 different languages.

This final token pool (dictionary) constitutes feature space for the data. Features are

[2] https://code.google.com/p/stop-words/

weighted by their term counts directly. In other words, by finding out the count of each stem from the pool for each retweet, these counts were turned into separate feature vectors which contain 1862 features in total.

In order to select significant features, information gain method (described in Section 3.3.2) on Weka is applied and the large feature space is reduced into 800 features (number of best features is obtained as 800 as it is shown to provide best accuracy in [9]). Finally, the dataset is named as EmoDS-2. As an example, a sample EmoDS-2 including 4 instances is shown in Table 4.3.

Table 4.3: Short View of EmoDS-2

Usernames	**Feature Vector Values**	**Labels**
@gioselyn_4	<1, 1, 1, 1, 1, 0, 0, 1, 1, ...>	sadness
@Ersiyn	<0, 0, 0, 1, 0, 0, 1, 1, 0, ...>	surprised
@Mukremin1973	<0, 0, 0, 0, 1, 0, 0, 2, 0, ...>	fear
@Feneristcom	<0, 0, 0, 0, 0, 0, 0, 1, 0, ...>	joy

4.5 Relationships Generation

4.5.1 Synthetic Relationships Generation

For the *only positive sentiment users*, a relationship data (244.104 instances) is generated for the SentDS dataset experiments. In this type of relationship, nothing but each positive sentiment user is matched with another positive sentiment user.

For generating *all users relationships*, "similar users hold similar sentiments" (homophily principle (Chapter 1, Section 1.1)) is assumed initially. Each user name with positive sentiment is randomly matched with each other and the same process is applied for the users with negative sentiment. Finally, under this setting, each user has at least one similar sentiment friend (neighbour). Also, each relationship is assumed as equal and their weights are set as 1.

Aiming to use in the collective classification experiments on EmoDS-1 dataset, four types of synthetic relationships are generated. Each of them is described as:

Relationship Generation 1 *(One-to-one Homophily Relations)*: For generating user

relationships, "similar users hold similar emotions" (homophily principle) are assumed initially. Each user with anger feeling is randomly matched with the other users having the same feeling and the same process is applied for the users with each of the other five feelings. Then, all matching users are joined together. Finally, in this setting, each user has at exactly one similar emotional friend (neighbour). Namely, all users who share the same feeling are not friends. Each relationship is assumed as equal and their weights are set as 1.

For the experiments, whole relationship data are divided into four parts that include 25% (1400 instances), 50% (2800 instances), 75% (4200 instances), 100% (5600 instances) relationship information in order.

Relationship Generation 2 *(Simple Anger-Joy Interaction)*: For simplicity, randomly chosen 1 anger labeled user is matched with 14 users whose emotions are spread such as: 2 anger, 2 disgust, 2 fear, 2 sadness, 2 surprise and 4 joy.

In that way, it can be asked how an angry user could be affected by his/her intensive joy friends. Training data is also reduced into 15 instances (1 user + 14 friends, their tweets and emotions). Each relationship is assumed as equal and their weights are set as 1.

Relationship Generation 3 *(Anger-Joy and Sadness-Joy Interaction)*: Similar to the second case of relationship generation, each of the angry users (800 in total) is matched with 14 friends who with the same emotion distribution (2 anger, 2 disgust, 2 fear, 2 sadness, 2 surprise and 4 joy). 11.158 user relationships are created in total.

For the crosscheck of experiment results, again each of the sad (800 in total) users are matched with 14 friends whose emotions are spread such as: 2 anger, 2 disgust, 2 fear, 2 sadness, 2 surprise and 4 joy. Due to the equal label separation, it is expected that experiment results remain same with the former.

Relationship Generation 4 *(More Realistic Random Relations)*: In the last phase of relationship generation, more realistic relations are created between users as:

Each user can have 30 friends at maximum and they are randomly matched with their friends (i.e. followers) regardless of their emotional labels. Thus, each user could have different number of relationships with different emotions.

It is named as *more realistic relationships-1*. Then, the same process is repeated provided that each user can have 120 friends at maximum. This second version of network is named as *more realistic relationships-2* in related experiment results.

4.5.2 Realistic Relationships Generation

In order to construct a network for EmoDS-2, by following interaction ways that are described in Section 3.1.1, all realistic friendship relations are extracted from the each retweet which contains RT flags and @ mentions. Self-edges caused by self-retweets and relations that are not unique (i.e. either one of the related usernames are not located as an instance in EmoDS-2) are discarded. At the end, 606 relationships are generated and they are visualized using open graph visualization platform, Gephi [3]. Generated relationships are presented in Figure 4.2.

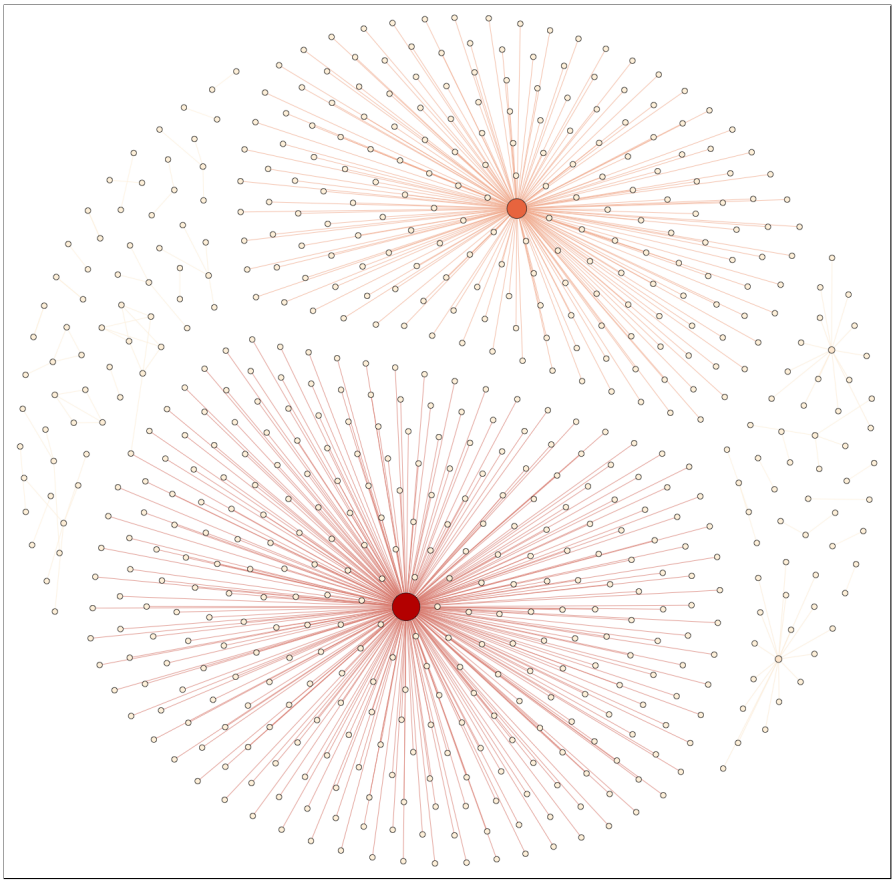

Figure 4.2: Visualisation of All Relationships Graph

[3] http://gephi.github.io

In addition, as a snapshot of a partial graph, 30 user nodes, their relationships and labels are illustrated on Figure 4.3. As shown, a sparse graph is formed.

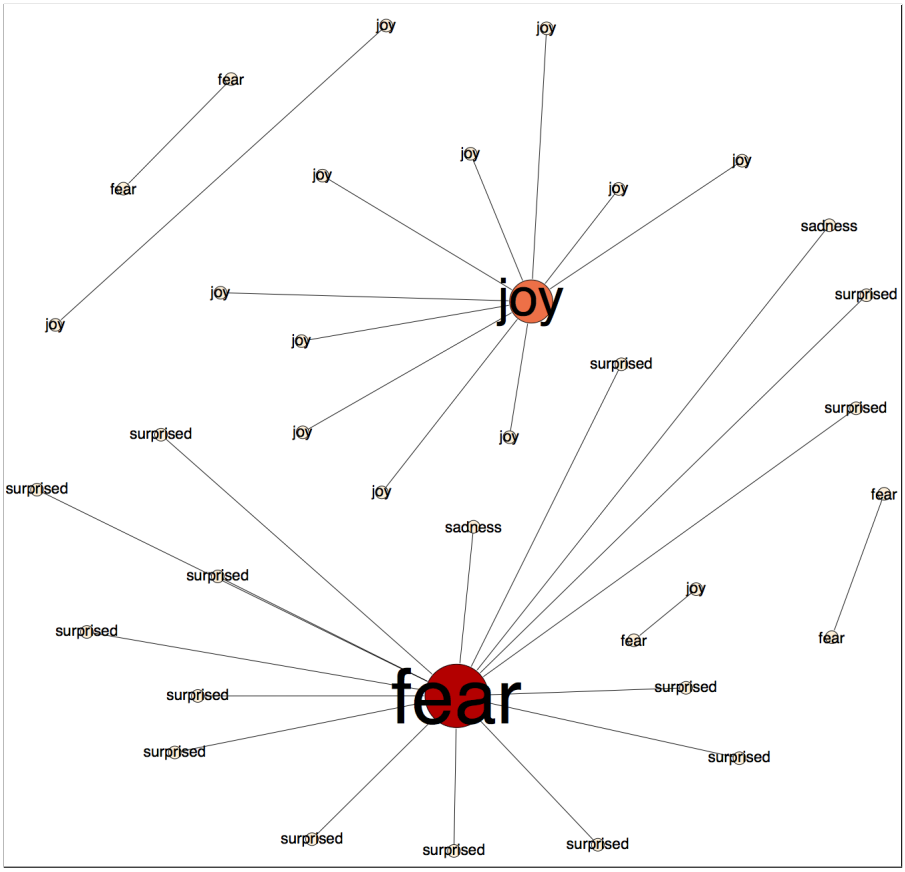

Figure 4.3: 30 Users with Relationships and Labels Graph

4.6 Classification

SentDS and EmoDS-1 datasets are used for collective classification only to see the preliminary behavior on synthetically constructed networks. Collective classification algorithms available in Netkit are applied on these data directly.

Classification is done on two categories on EmoDS-2 dataset. First one is the SVM classification as a baseline algorithm for comparison purpose. Latter is the collective classification with realistically generated relationship information. Since, neighbor features are under evaluation for the related experiments, user retweets are preprocessed for possible noise eliminations and represented as feature vectors. Large feature space is also reduced with information gain method. Then, Netkit's own collec-

tive algorithms, also expanded with proposed relational classifier (Section 4.6.1.1), are applied on EmoDS-2 precisely.

4.6.1 Collective Classification

Collective classification algorithms are applied with Netkit-SRL tool (Section 3.4.3). Different combinations of relational classifiers and inference methods are employed. Also, relational classifier's aggregation options are changed for some cases to understand Netkit's readily available relational classifier's behavior.

4.6.1.1 Proposed Network-Only Bayes Relational Classifiers

McDowell and Aha [21] investigate the neighboring attributes' contribution while building relational models and estimating it with collective inferencing on partially labeled networks. They propose a probabilistic relational model for bringing neighbor attributes and labels together. Results show that using both neighbor attributes and labels on building relational model, often produces the best accuracy.

However, this study is tested under some small sparsely-labeled networked datasets. Moving beyond this approach to fully-labeled sentimental social networks, it can be asked whether the obtained results are consistent or not for this supervised setting. Inspired from this idea, Netkit's network only bayes relational classifier (*nobayes*, Section 3.2.2.3.2) is expanded with adding neighbor features information into process and implemented in Netkit environment.

The proposed algorithm is started with a simple probabilistic assumption that each neighbor's features are conditionally independent. Because attributes are represented as feature vectors, their cosine similarities between its neighbors are computed for prior probability calculations, rather than simply counting. Then, these scores are used in the objective function given in Equation 4.1 as a new variable.

$$P(N_i|c) = \frac{1}{Z} \prod_{v_j \in N_i} P(c_j = c_j^{'}|c_i = c)^{weight_{i,j}} \times P(simscore_j|c_i = c)^{weight_{i,j}} \quad (4.1)$$

In Equation 4.1, Z is standard normalization variable and $simscore_j$ is the cosine similarity score observed at node *j*.

Relational classification based on Equation 4.1 is called as Network-Only Bayes-VectorSimilarity classifier, *nobayes-vecsim* for short. Pseudo-code for Netkit-SRL implemented proposed *nobayes-vecsim* relational classifier is shown on Algorithm 3. Hence, while estimating the proposed objective function with collective inferencing method, neighbor's features also become valuable. On the other hand, if neighbor feature vectors are disparate with unknown labeled user's feature vector, it is expected not too much increase on classification accuracies.

Since this thesis approach is user-centric, emotion data EmoDS-2 have unique user instances where each of them has only one unique *retweet*. Consecutively, each neighbor node has only one useable feature vector for the proposed relational classifier *nobayes-vecsim*.

Another version of this relational classifier is also implemented as *nobayes-avgsim*. It takes account of the case that a user can have different number of *retweets* under different emotion classes. However, Netkit is only configured to use unique user instances and does not allow to use the same username (seen as *key* attribute) under different classes.

For this case, a different implement strategy is employed as follows: cosine similarities between each neighbor user's feature vectors are calculated and the average is taken at the end. Then, averaged similarity score for each node is fed into relational classifier's Equation 4.1 externally. Pseudo-code for Netkit-SRL implemented proposed *nobayes-avgsim* relational classifier is shown on Algorithm 4.

From the point of EmoDS-2 data view, not only class labels of their friends but also their attributes are taken into consideration to understand whether the expression of emotions are consistent with their friends or not. In other words, it is expected to see the effect on collective classification of user emotions when utilizing their friends label and local attribute information together.

Algorithm 3 Pseudo-code of Proposed Nobayes-Vecsim Relational Classifier

function InduceRelationalModel

for all $v_i \in T$ **do**

$c\_prior\_prob\_vec \leftarrow v_i$'s class value counts // find $P(c)$

for all $v_j \in N_i$ **do**

$c\_nbor\_prob\_vec \leftarrow v_j$'s class value counts powered with edge weights // find $P(c_j = c_j^{'}|c_i = c)^{weight_i,j}$

$simscore\_nbor\_prob\_vec \leftarrow v_j$'s cosine similarity scores powered with edge weights // find $P(simscore_j|c_i = c)^{weight_i,j}$

end for

end for

Normalize each vector

end function

function ApplyEstimation // for the inference phase, estimate Equation 4.1

$estimation\_prob\_vec := \{\}$

for all $v_i \in U$ **do**

$known\_prob\_vec \leftarrow c\_prior\_prob\_vec$

for all $v_j \in N_i$ **do**

$known\_prob\_vec$ *= $c\_nbor\_prob\_vec * simscore\_nbor\_prob\_vec$

end for

end for

$estimation\_prob\_vec \leftarrow known\_prob\_vec$

return $estimation\_prob\_vec$ as v_i's class estimations

end function

Algorithm 4 Pseudo-code of Proposed Nobayes-Avgsim Relational Classifier

for all $v_i \in G$ **do**

$featvec\_v_i \leftarrow v_i$'s features

for all $v_j \in N_i$ **do**

$featvec\_v_j \leftarrow v_j$'s features

Find cosine similarity scores between $featvec\_v_i$ and for each $featvec\_v_j$

Take the average as $avg\_simscore\_values$

end for

end for

return $avg\_simscore\_values$ for all $v_i \in G$

function InduceRelationalModel

for all $v_i \in T$ **do**

$c\_prior\_prob\_vec \leftarrow v_i$'s class value counts // find $P(c)$

for all $v_j \in N_i$ **do**

$c\_nbor\_prob\_vec \leftarrow v_j$'s class value counts powered with edge weights // find $P(c_j = c_j^{'} | c_i = c)^{weight_i,j}$

$simscore\_nbor\_prob\_vec \leftarrow v_j$'s cosine similarity scores from $avg\_simscore\_values$ powered with edge weights // find $P(simscore_j | c_i = c)^{weight_i,j}$

end for

end for

Normalize each vector

end function

function ApplyEstimation // for the inference phase, estimate Equation 4.1

$estimation\_prob\_vec := \{\}$

for all $v_i \in U$ **do**

$known\_prob\_vec \leftarrow c\_prior\_prob\_vec$

for all $v_j \in N_i$ **do**

$known\_prob\_vec$ *= $c\_nbor\_prob\_vec * simscore\_nbor\_prob\_vec$

end for

end for

$estimation\_prob\_vec \leftarrow known\_prob\_vec$

return $estimation\_prob\_vec$ as v_i's class estimations

end function

CHAPTER 5

EXPERIMENTS

In this section, conducted experimental analysis and the obtained results are presented. Each of the three Twitter data sets is tested under collective classification framework, Netkit, with its generated synthetic or realistic relationships. While processing, important relational classifier-inference method combinations are analyzed closely.

Experiments are evaluated with 10-fold cross validation. It splits the nodes into 10 equal-sized sets and performs 10 evaluations runs where it holds out one of the sets (test set) and trains on the rest. It then predicts the test set and finally gives averaged accuracy performance.

In some experiments, relational classifiers are configured with *-All* aggregation option to see the algorithm's consideration on neighbor's all features. However, this option is dependent only to Netkit's own implementations and it could only be set as *-ClassOnly* for most of the relational classifiers. In addition, proposed relational classifier algorithms (Section 4.6.1.1) are experimented on EmoDS-2 dataset with realistic relationships.

5.1 Dataset Descriptions

5.1.1 Sentiment140 Twitter Dataset (SentDS)

Original SentDS dataset contains 1.600.000 instances which has been processed so that the emoticons are extracted. Also, it's in a regular CSV format and includes 6

fields as follows:

1. the polarity of the tweet (0 = negative, 4 = positive)
2. the id of the tweet
3. the date of the tweet
4. the query. If there is no query, then this value is NO_QUERY.
5. the user that tweeted
6. the text of the tweet in English language

5.1.2 Emotional Twitter Datasets

There are two emotional datasets which have used in this thesis. They are described under the following subsections respectively.

5.1.2.1 Demirci's Emotion Dataset (EmoDS-1)

Original emotion training dataset contains 4800 instances. Also, it's in a regular CSV format and includes 3 fields as follows:

1. the id of the user (there are 4800 unique users so index is from 1 to 4800)
2. the text of the tweet in Turkish language
3. the emotion label of the tweet

All training data are split into equal class labeled tweets and indexed as seen on Table 5.1.

Table 5.1: EmoDS-1 Class Label Distribution due to User ID Ranges

User ID Ranges	Class Labels
0-799	anger
800-1599	disgust
1600-2399	fear
2400-3199	joy
3200-3999	sadness
4000-4799	surprise

5.1.2.2 Emotion Dataset Collected within This Study (EmoDS-2)

However, since this work's approach is user-centric, duplicate usernames are eliminated from the whole data. Finally, there are 6841 instances (unique usernames are regarded as key field) which each contains three fields named as Usernames, (Re) Tweets, Labels. Class distributions could be seen on Table 5.2.

Table 5.2: EmoDS-2 Class Label Distribution due to Instance Counts

Class Labels	Instance Counts
anger	1118
disgust	1140
fear	1145
joy	1191
sadness	1121
surprise	1126

5.2 Experimental Analysis on Collective Classification for Sentiment Analysis

5.2.1 Collective Classification and Dataset Size Performance Results

In order to analyze the collective classification performance under different data sizes for a simpler task of binary classification, i.e. sentiment analysis, all SentDS data is divided into 10 parts which contains increasing amounts of instance as 10% (65.977 instances), 20% (131.955 instances), 30% (197.933 instances), 40% (263.911 instances), 50% (329.889 instances), 60% (395.867 instances), 70% (461.845 instances), 80% (527.823 instances), 90% (593.801 instances),100% (659.773 instances) in order.

Then, each part is experimented on Netkit with 21 different collective classification configurations. Each of these configuration's components are explained in detail in Section 3.2.2.3. They are included in Table 5.3, Table 5.4 and Table 5.5 together with their descriptions. By the way, in all cases mentioned in Table 5.3, the local classifier is applied, and then the relational classifier is applied only once.

Table 5.3: Collective Classification Configurations with Their Descriptions

Configuration	**Description**
WVRN-Null Inference	Collective classification using Weighted-Vote Relational Neighbour Classifier.
PRN-Null Inference	Collective classification using Probabilistic Relational Neighbour Classifier.
CDRN-NORM-COS-Null Inference	Collective classification using Class-Distribution Relational Neighbour Classifier.
NOBAYES-Null Inference	Collective classification using Network-Only Bayes Relational Classifier.
NOLB-LR-DISTRIB-Null Inference	Collective classification using Network-Only Link-Based Relational Classifier.
NAIVEBAYES-Null Inference	Collective classification using Multinomial Naive Bayes as a relational classifier wrapped from Weka.
J48-Null Inference	Collective classification using Decision Tree as a relational classifier wrapped from Weka.

Table 5.4: Collective Classification Configurations with Their Descriptions

Configuration	**Description**
WVRN-Iterative	Collective classification using Iterative Classification inference method with Weighted-Vote Relational Neighbour Classifier.
PRN-Iterative	Collective classification using Iterative Classification inference method with Probabilistic Relational Neighbour Classifier.
CDRN-NORM-COS-Iterative	Collective classification using Iterative Classification inference method with Class-Distribution Relational Neighbour Classifier.
NOBAYES-Iterative	Collective classification using Iterative Classification inference method with Network-Only Bayes Relational Classifier.
NOLB-LR-DISTRIB-Iterative	Collective classification using Iterative Classification inference method with Network-Only Link-Based Relational Classifier.
NAIVEBAYES-Iterative	Collective classification using Iterative Classification inference method with Multinomial Naive Bayes as a relational classifier wrapped from Weka.
J48-Iterative	Collective classification using Iterative Classification inference method with Decision Tree as a relational classifier wrapped from Weka.

Table 5.5: Collective Classification Configurations with Their Descriptions

Configuration	**Description**
WVRN-RelaxLabel	Collective classification using Relaxation Labeling inference method with Weighted-Vote Relational Neighbour Classifier.
PRN-RelaxLabel	Collective classification using Relaxation Labeling inference method with Probabilistic Relational Neighbour Classifier.
CDRN-NORM-COS-RelaxLabel	Collective classification using Relaxation Labeling inference method with Class-Distribution Relational Neighbour Classifier.
NOBAYES-RelaxLabel	Collective classification using Relaxation Labeling inference method with Network-Only Bayes Relational Classifier.
NOLB-LR-DISTRIB-RelaxLabel	Collective classification using Relaxation Labeling inference method with Network-Only Link-Based Relational Classifier.
NAIVEBAYES-RelaxLabel	Collective classification using Relaxation Labeling inference method with Multinomial Naive Bayes as a relational classifier wrapped from Weka.
J48-RelaxLabel	Collective classification using Relaxation Labeling inference method with Decision Tree as a relational classifier wrapped from Weka.

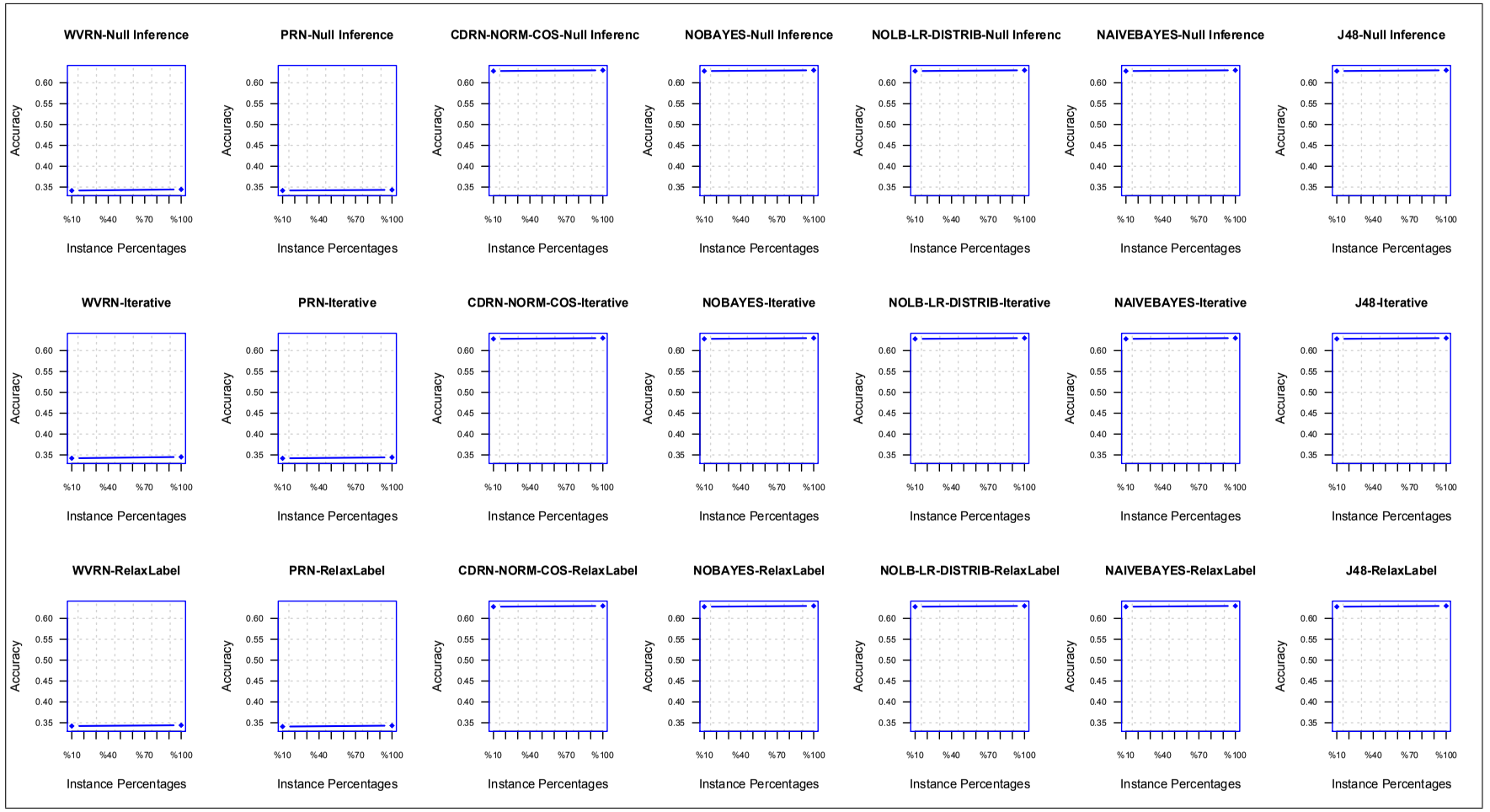

Figure 5.1: Collective Classification Accuracy vs. Dataset Size Results of SentDS Dataset

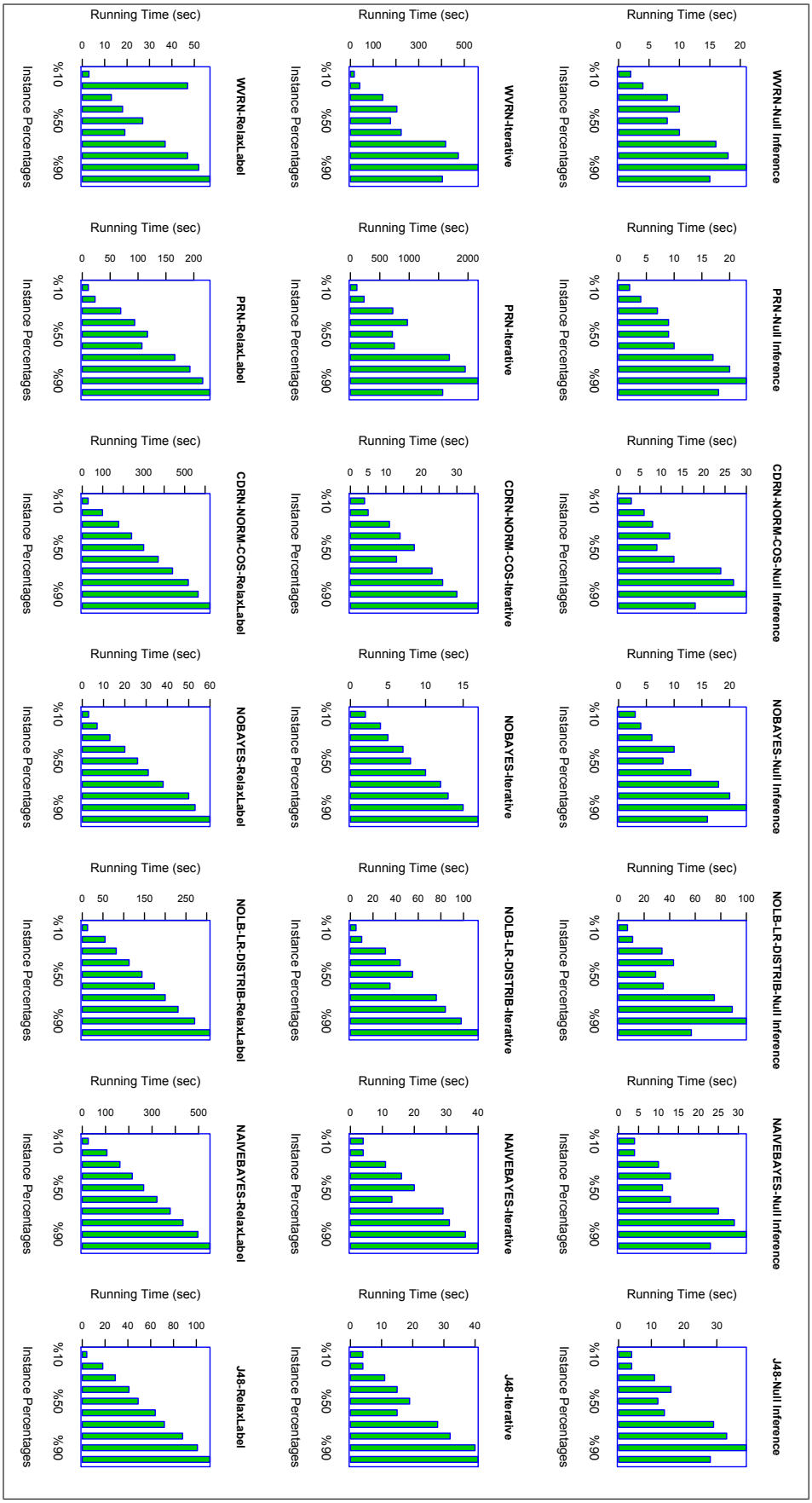

Figure 5.2: Collective Classification Running Time vs. Dataset Size Results of SentDS Dataset

Obtained accuracy and time performances can be seen in Figure 5.1 and Figure 5.2. Weighted-vote relational neighbour classifer and probabilistic relational neighbour classifer are almost achieved 30% less accuracy performance than the others. Since these algorithms are not designed to build a relational model before the inference phase, it does not matter how big the training data is. Also for them, iterative classification is not a good choice for dealing with uncertainties of neighbor's labels due to its running time performances.

As seen in Figure 5.1 and Figure 5.2, all of the collective classification configurations with link-based relational classifiers reach almost the same accuracy results around 60%. Especially, Network-Only Bayes relational classifier with iterative classification inference method combination behaves generally well according to its time performances.

NoBayes-Iterative classification results can be examined in closer in Figure 5.3 and Figure 5.4. While running on whole data instances, it achieves the same best accuracy result with other configurations in about ~17 seconds. It rapidly converges the stable class membership estimations for the unknown labeled nodes during iterative inferencing. Since, there is no relationship information, NoBayes relational classifier induces its model directly from local classifier's prior class distributions at the training phase.

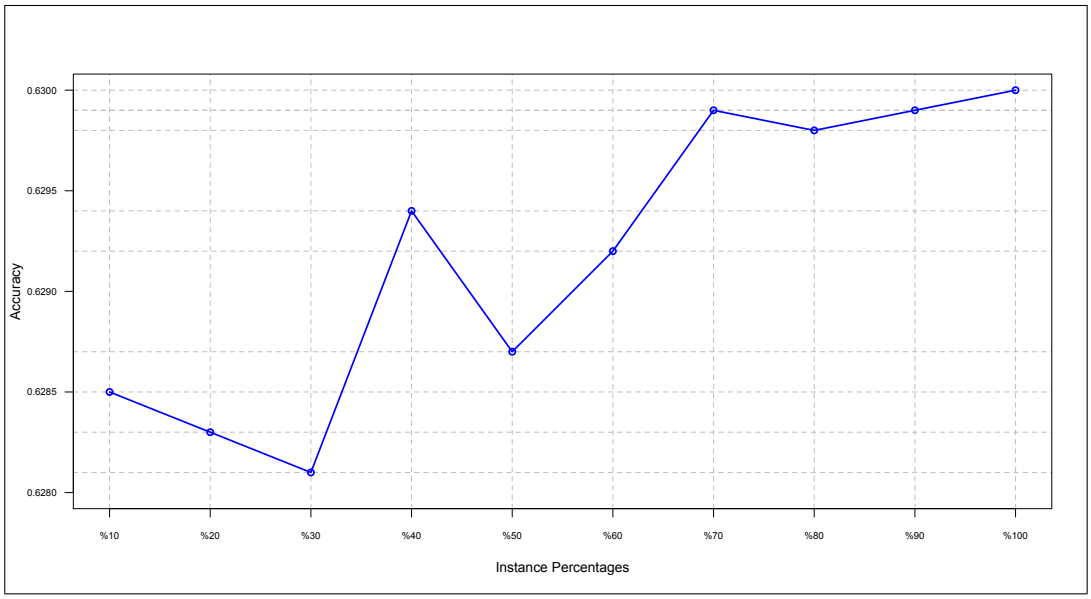

Figure 5.3: NoBayes-Iterative Classification Accuracy vs. Dataset Size Results of SentDS Dataset

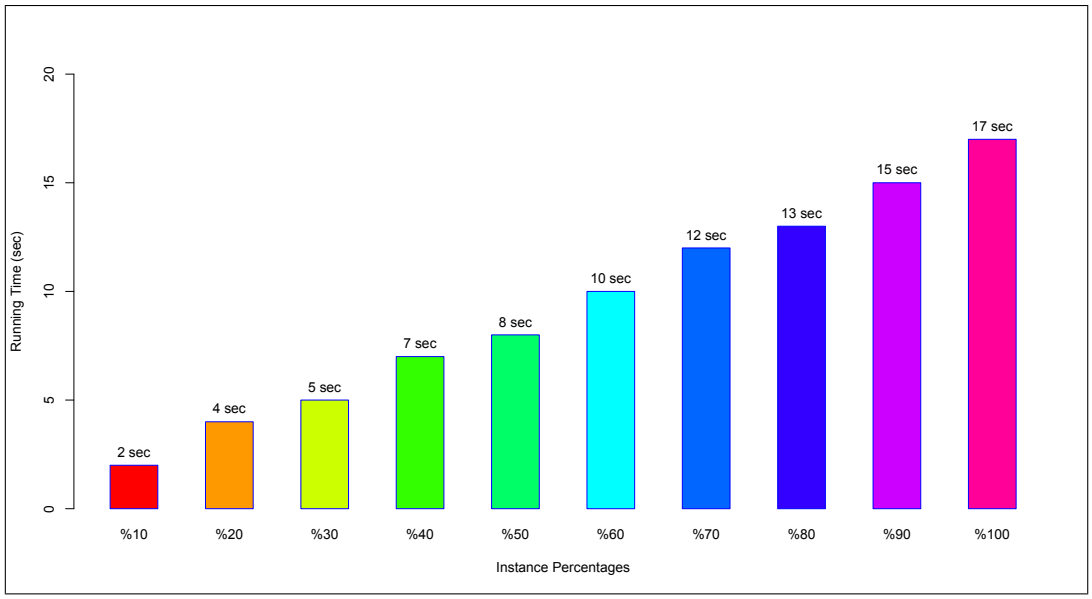

Figure 5.4: NoBayes-Iterative Classification Running Time vs. Dataset Size Results of SentDS Dataset

5.2.2 Collective Classification with Synthetic Relationships Results

Each collective inferencing method-relational classifier configuration are run on same training data with generated two different types of synthetic relationship information.

In the first phase, for understanding how useful relationship information contributes to collective classification task, *only positive sentiment users* relationship data are added to the process. Link-based relational classifiers (nobayes, cdrn-norm-cos, nolb-lr-distrib) combined with iterative inferencing has leaded great accuracies in short time. Results can be seen in Table 5.6 and Table 5.7.

Table 5.6: Collective Classification with Only Positive Sentiment User Relationships Accuracy Results

	wvrn	**prn**	**cdrn-norm-cos**	**nobayes**	**nolb-lr-distrib**
Null Inference	0.5255	0.5258	0.9999	0.9999	0.9999
Iterative Classification	0.5272	0.5263	0.9999	0.9999	0.9999
Relaxation Labeling	0.5271	0.5276	0.9999	0.9999	0.9999

Table 5.7: Collective Classification with Only Positive Sentiment User Relationships Running Time (sec) Results

	wvrn	**prn**	**cdrn-norm-cos**	**nobayes**	**nolb-lr-distrib**
Null Inference	21	23	24	19	183
Iterative Classification	552	2249	33	23	194
Relaxation Labeling	64	231	302	59	298

In the second phase, *all users relationships* data is divided into four parts that includes 25% (164.944 instances), 50% (329.886 instances), 75% (494.829 instances), 100% (659.773 instances) relationship information in order. Then, these four different amount of relationship information are experimented respectively. Obtained accuracy and time performances can be seen in Figure 5.5 and Figure 5.6.

On each step, adding more relationship information is increased final accuracies significantly. On the other hand, time consumption could be change for different combinations except for some of them. However, network only bayes classifier (nobayes) – iterative classification combination results with the best accuracy among all analyzed classifiers. This coheres with the Netkit's own published results in [20].

5.3 Experimental Analysis on Collective Classification for Emotion Analysis

5.3.1 Experimental Analysis on EmoDS-1 Dataset

5.3.1.1 Collective Classification with Synthetic Relationships Results

Each collective inferencing method-relational classifier configuration are run on the same training data under generated four different types of synthetic relationship information. For each of the relationship types, base performances were tested on with only the training data (no relationship information) for comparison purposes.

For understanding how much *one-to-one homophily relationship* information contributes to collective classification task, relationship data are added to the process at varying percentages. Link-based relational classifiers (nobayes, cdrn-norm-cos etc.) combined with iterative inferencing method has leaded great accuracies in short time.

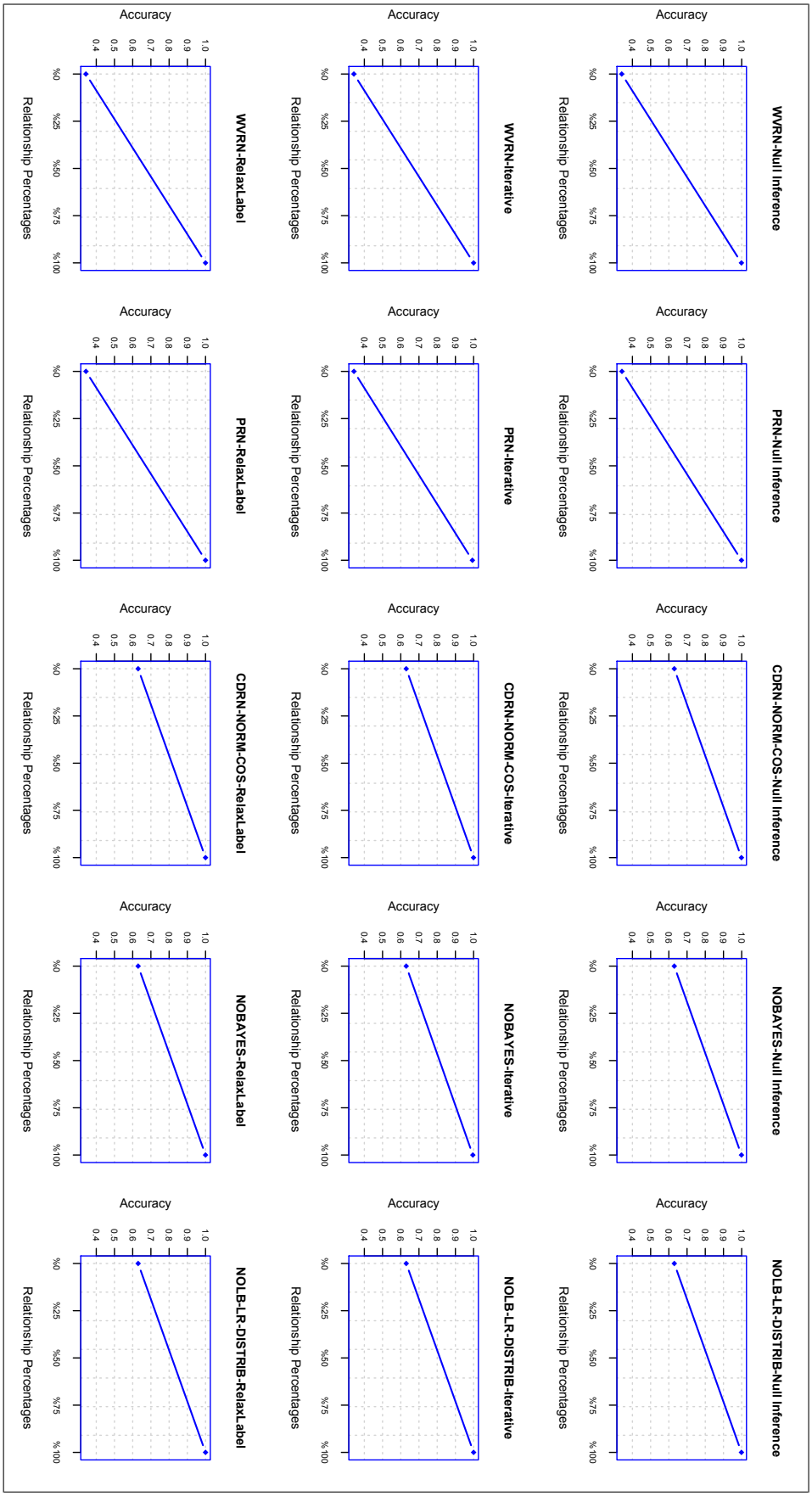

Figure 5.5: Collective Classification Accuracy vs. Relationship Size Results of SentDS Dataset

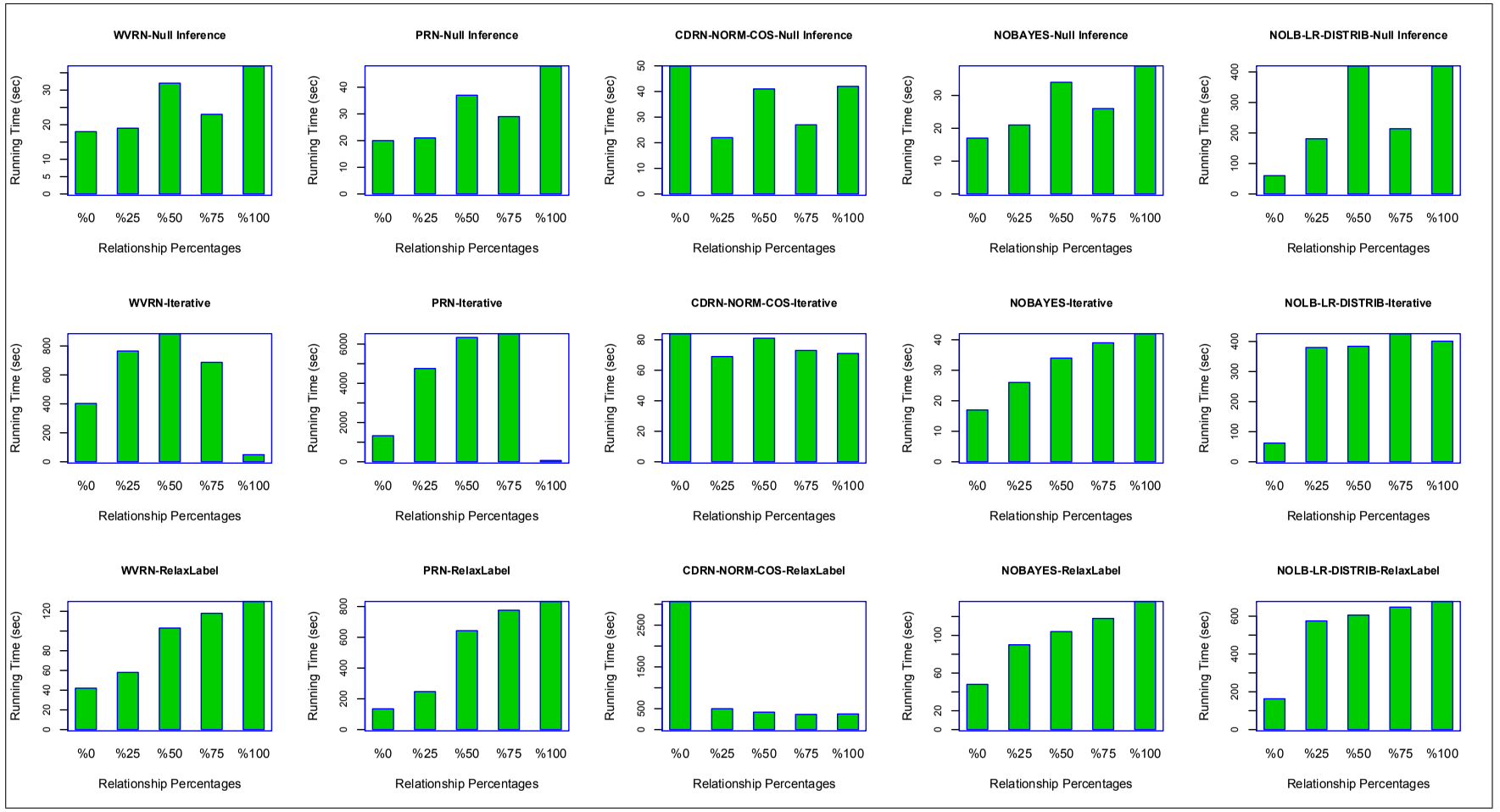

Figure 5.6: Collective Classification Running Time vs. Relationship Size Results of SentDS Dataset

As more amount of relationship is included in the data set, higher accuracy results are obtained, as expected. This coheres with the Netkit's own published results in [20]. Obtained accuracy and time performances can be seen in Figure 5.7 and Figure 5.8.

For the *simple anger-joy emotion interactions* experiments, relational classifiers (aggregation with only class values of neighbours) wrapped from WEKA – iterative classification combinations are leaded considerable accuracies in short time. Results are on Table 5.8 and Table 5.9.

Table 5.8: Collective Classification with Simple Anger-Joy User Relationships Accuracy Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.15	0.1	0.05	0.01	0.3	0.2	0.01
Iterative Classification	0.2	0.1	0.2	0.01	0.25	0.25	0.1
Relaxation Labeling	0.1	0.15	0.25	0.01	0.25	0.25	0.2

Table 5.9: Collective Classification with Simple Anger-Joy User Relationships Running Time (msec) Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	219	15	31	20	185	68	32
Iterative Classification	10	7	26	10	31	33	15
Relaxation Labeling	18	58	44	15	65	113	17

On the other hand, when relational classifiers include aggregation option on all attributes of neighbours, naive bayes relational (wrapped from Weka) classifier has got the best accuracy result (40%) as shown on Table 5.10 and Table 5.11.

Table 5.10: Collective Classification with Simple Anger-Joy User Relationships Accuracy Results (aggr. -All)

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.1	0.15	0.1	0.01	0.15	0.25	0.15
Iterative Classification	0.15	0.15	0.15	0.01	0.15	0.25	0.25
Relaxation Labeling	0.2	0.1	0.2	0.01	0.25	0.4	0.15

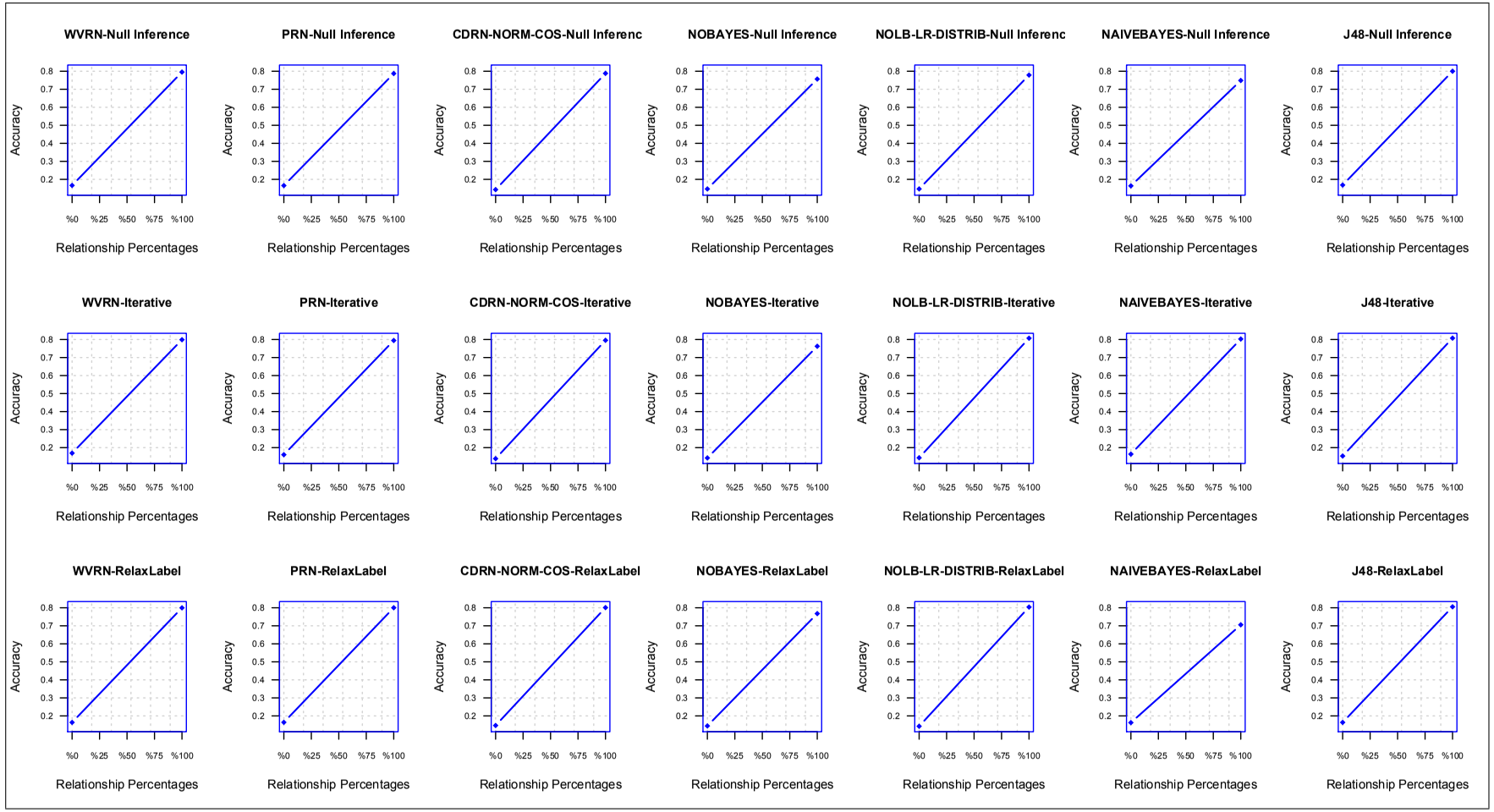

Figure 5.7: Collective Classification Accuracy vs Relationship Size Results of EmoDS-1 Dataset

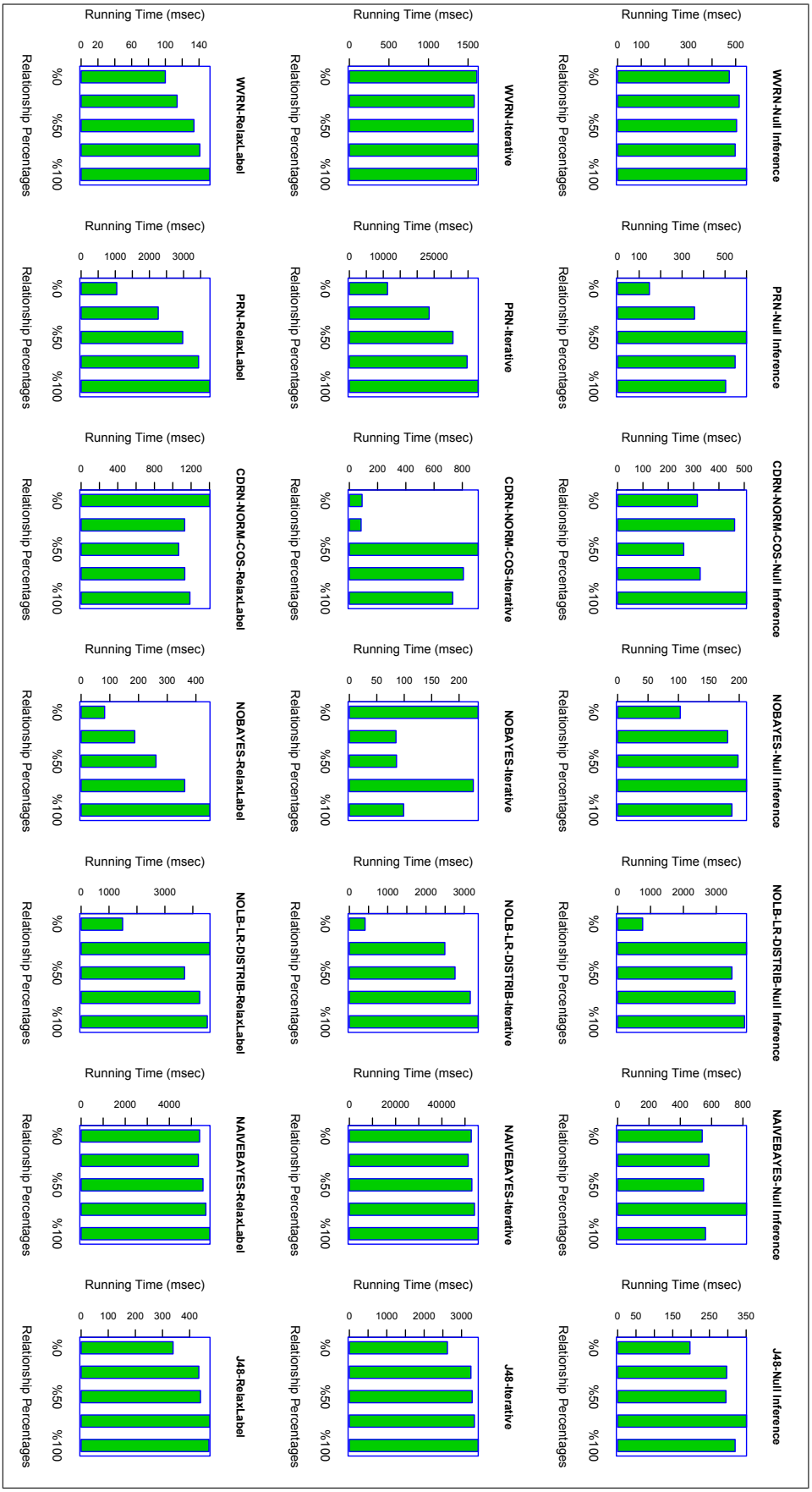

Figure 5.8: Collective Classification Running Time vs Relationship Size Results of EmoDS-1 Dataset

Table 5.11: Collective Classification with Simple Anger-Joy User Relationships Running Time (msec) Results (aggr. -All)

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	117	15	76	42	288	97	45
Iterative Classification	9	6	45	23	70	51	26
Relaxation Labeling	14	78	31	19	104	136	19

For the *anger-joy* and *sadness-joy emotion interactions* experiments, network only bayes (nobayes) - iterative inferencing is achieved best results. Also, while training and relation data get much bigger, accuracies increase significantly. Results are shown on Table 5.12, Table 5.13, Table 5.14 and Table 5.15 respectively.

Table 5.12: Collective Classification with Anger-Joy User Relationships Accuracy Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.09	0.103	0.332	0.227	0.293	0.306	0.291
Iterative Classification	0.118	0.105	0.328	0.260	0.313	0.332	0.316
Relaxation Labeling	0.117	0.117	0.327	0.244	0.304	0.332	0.318

Table 5.13: Collective Classification with Anger-Joy User Relationships Running Time (sec) Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.56	0.46	0.5	0.26	3	0.63	0.24
Iterative Classification	2	42	13	0.12	3	0.44	0.17
Relaxation Labeling	0.18	4	1	0.52	4	7	0.41

Table 5.14: Collective Classification with Sadness-Joy User Relationships Accuracy Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.098	0.108	0.342	0.233	0.296	0.305	0.290
Iterative Classification	0.120	0.114	0.331	0.225	0.313	0.333	0.315
Relaxation Labeling	0.113	0.113	0.331	0.231	0.318	0.333	0.311

Table 5.15: Collective Classification with Sadness-Joy User Relationships Running Time (sec) Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.57	0.39	0.53	0.25	3	0.59	0.25
Iterative Classification	2	33	13	0.12	3	0.46	0.17
Relaxation Labeling	0.19	4	1	0.51	4	7	0.43

Finally, two more realistic random relationships were tested in order. For the first network (*more realistic relationships-1*, each user has 30 friends at maximum), nobayes – iterative classification has achieved the fastest result in ~0.35 seconds (353 milliseconds) as seen on Table 5.17. For the second network (*more realistic relationships-2*, each user has 120 friends at maximum), again same combination has achieved the fastest result in ~1 seconds (1230 milliseconds) as seen on Table 5.19.

As expected, amount of relation is directly proportional to running time. Whereas, same idea is not applicable for the accuracies as seen on Table 5.16 and Table 5.18. This collective classification combination could be improved by healing relational classifier algorithm's prediction capability. For this purpose, thesis proposed relational classifier (*nobayes-vecsim*) is focused on the network only bayes algorithm.

Table 5.16: Collective Classification with More Realistic Relationships-1 Accuracy Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.177	0.173	0.159	0.150	0.157	0.164	0.171
Iterative Classification	0.181	0.170	0.173	0.144	0.164	0.165	0.163
Relaxation Labeling	0.168	0.175	0.170	0.152	0.161	0.172	0.170

Table 5.17: Collective Classification with More Realistic Relationships-1 Running Time (sec) Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.69	0.51	0.52	0.44	3	0.71	6
Iterative Classification	5	54	0.35	0.35	5	43	17
Relaxation Labeling	0.65	7	2	4	4	5	7

Table 5.18: Collective Classification with More Realistic Relationships-2 Accuracy Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.152	0.158	0.169	0.144	0.152	0.172	0.171
Iterative Classification	0.142	0.167	0.162	0.155	0.165	0.171	0.172
Relaxation Labeling	0.158	0.165	0.167	0.151	0.163	0.173	0.164

Table 5.19: Collective Classification with More Realistic Relationships-2 Running Time (sec) Results

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	1	1	1	1	4	1	6
Iterative Classification	20	135	1	1	12	68	44
Relaxation Labeling	3	14	5	18	7	9	10

5.3.2 Experimental Analysis on EmoDS-2 Dataset

5.3.2.1 SVM Classification Result

As a baseline algorithm that is selected as best method from study in [9], libSVM algorithm on WEKA (Section 3.4.2) with default values (radial basis kernel function, 10-fold cross validation) is applied onto EmoDS-2 and leaded to 87.9167% accuracy. Also, it helps to gain insights on the reliability of constructed feature vectors.

5.3.2.2 Collective Classification with Realistic Relationships Results

For the purpose of seeing the defined thesis problem clearly from the real twitter world perspective, each collective inferencing method-relational classifier configuration are run on emotion data with generated realistic relationship informations. Also, proposed relational classifier, *nobayes-vecsim* and its another version, *nobayes-avgsim* are taken into consideration.

As described on Section 4.6.1.1, according to the proposed relational classifier's na-

ture, it looks for each user's each neighbour's feature vector similarities, multiples edge weights with this value and uses it as a new variable in probability function calculation while inferencing. Differently, it is provided to add neighbour's attributes into process by changing aggregation option to "All" for the other configurations if it is already implemented in Netkit-SRL by default. In this way, relational classifier not only thinks class labels of its neighbours but also aggregates on attributes of them and adds as a new feature for prediction.

In the first phase, each of collective classification combinations are tried with no relationship information and "All" aggregation option. Results are shown on Table 5.20 and Table 5.21.

Table 5.20: Collective Classification with No Relationships Accuracy Results (aggr. -All)

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	***no-bayes-vecsim***	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.165	0.173	0.174	0.174	0.174	0.174	0.912	0.986
Iterative Classification	0.171	0.170	0.174	0.174	0.174	0.174	0.912	0.986
Relaxation Labeling	0.165	0.170	0.174	0.174	0.174	0.174	0.911	0.986

Table 5.21: Collective Classification with No Relationships Running Time (sec) Results (aggr. -All)

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	***no-bayes-vecsim***	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	2	1	14	2	1	46	170	749
Iterative Classification	3	17	15	1	1	47	321	740
Relaxation Labeling	1	3	162	1	1	125	8822	950

In the second phase, to see the relationship information effect on user's emotions, each of collective classification combinations are tried with all relationship information and "All" aggregation option. Results are shown on Table 5.22, Table 5.23, Table 5.24 and Table 5.25 respectively.

Table 5.22: Collective Classification with All Relationships Accuracy Results (aggr. -All)

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	***no-bayes-vecsim***	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.187	0.192	0.190	0.207	0.206	0.220	0.909	0.985
Iterative Classification	0.187	0.195	0.189	0.207	0.211	0.223	0.909	0.986
Relaxation Labeling	0.192	0.186	0.190	0.208	0.208	0.220	0.909	0.985

Table 5.23: Collective Classification with All Relationships Running Time (sec) Results (aggr. -All)

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	***no-bayes-vecsim***	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	2	2	14	2	11	498	170	762
Iterative Classification	3	20	15	1	11	525	1031	787
Relaxation Labeling	1	3	173	1	11	711	8789	974

Table 5.24: Collective Classification with All Relationships Accuracy Results-2 (aggr. -All)

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	***no-bayes-avgsim***	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	0.190	0.188	0.193	0.213	0.211	0.237	0.908	0.987
Iterative Classification	0.194	0.195	0.190	0.217	0.210	0.231	0.909	0.986
Relaxation Labeling	0.195	0.189	0.195	0.214	0.208	0.235	0.909	0.985

Table 5.25: Collective Classification with All Relationships Running Time (sec) Results-2 (aggr. -All)

	wvrn	**prn**	**cdrn-norm-cos**	**no-bayes**	***no-bayes-avgsim***	**nolb-lr-distrib**	**naive-bayes**	**j48**
Null Inference	1	1	15	1	1	659	169	788
Iterative Classification	3	20	16	1	1	580	1559	823
Relaxation Labeling	1	2	181	1	1	880	8798	1026

As seen from the results, generated sparse user friendship relations are yielded in different ranges between users so that some users do not share the same feelings with their friends as opposite to homophily principle. This situation is to be observed by seeing small performance gain (~5%) results in collective classification accuracies.

Proposed relational classifier (*nobayes-vecsim*) leads to a small increase in accuracies. However, it consumes considerable amount of time. This situation is pretended to be healed with *nobayes-avgsim* relational classifier's running time. Actually, it is completely due to the implementation differences of two algorithms. In *nobayes-avgsim*, by calculating vector similarities before the classification time, almost the same accuracy results are obtained in less time.

On the other hand, Netkit-SRL is able to wrap some classifiers from WEKA as relational classifiers (Section 3.2.2.3.2) and these are leaded up to 98.6% significant accuracies. However, learning relational models takes long time (~13 minutes for iterative classification-j48 combination) for them. It can be seen that they are not affected from the relationship information too much and features reliability is much more important for them.

By looking from Netkit collective classification view, decision tree (j48) algorithm (Section 3.2.2.3.2) aggregate on related neighbor features collectively and add them as new attributes into unknown labeled node's feature space while estimating model by the inference method (iterative classification).

5.4 Overall Discussion of Experimental Results

SentDS dataset experiments show that the training data size is proportional to small amount of increase on prediction accuracies except from two relational algorithms that are weighted-vote relational neighbor (wvrn) and probabilistic relational neighbour (prn) classifiers. It is not important to have much data instance to predict unknown labels for them. They could be useful for less-data domains.

Network-only-bayes (nobayes) relational classifier combined with iterative classification comes to the fore with its time performances. It predicts on whole data in about

17 seconds. This situation is due to the nature of its algorithmic operations. Big data classification may be a good choice for this collective classification configuration.

Then, synthetic relationships experiments on SentDS data are held on two phases. In light of homophily principle, if there is a user who has at least one friend with positive sentiment, its unknown labeled is strongly predictable as positive. On the other hand, adding more relationship information is increased all accuracies significantly.

So, for the user relations, preserving homophily idea, it can be stated that the more relationship information is gathered, the better increase is obtained in collective classification. This situation is supported by the initial experiment results on EmoDS-1 dataset.

On EmoDS-1 data, user's emotion interactions are tested with synthetically generated relationships. Anger-joy and sadness-joy emotion pairs are experimented and consistent results show that user emotions can be affected from their friend's intensive emotions. Finally, two more realistic random relationships experiment results present that possible different bias on the emotions of their neighbors can harm prediction capabilities of user's unknown emotional labels. Also, amount of relationship is directly proportional to running time performances.

In the last phase, by seeing previous experiments as preliminary, EmoDS-2 dataset is tested with generated realistic relationships. It is investigated how previous obtained results are prevailing for the real Twitter world perspective. The proposed algorithm *nobayes-vecsim* and its other version, *nobayes-avgsim* are also tried with other combinations. As seen from the results, sparse relationships between users indicate that friends do not tend to share similar emotions with their connected users.

However, wrapped relational classifier from Weka, decision trees (j48), yield to significant accuracy results. Due to the reliability of instance features, attributes gain more importance then relationship information for that algorithm's nature. While aggregating on all features of the neighbors contribute to prediction performance too much, the same can not be said for running time performances.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this thesis, user emotions in social networks are aimed to be predicted in a collaborative setting. Different collective classification configurations are applied on related sentimental datasets with their kind of relationship information. Different results are obtained especially for the real world of Twitter.

It was expected to have strong correlations with *retweet* relationships on Twitter social network. The impact of that information is evaluated on the experiment results obtained. However, generated thesis Twitter network (*retweet*) relationship information is not very useful for collective classification task due to its sparse and different emotion distribution.

Text-based features (Turkish *retweets*) of users are turned into compact feature vectors by using language-specific tools and methods. Combining classical text mining techniques with collective classification leads great results especially for some relational algorithms. As in traditional machine learning methods, these algorithms benefit from reliable neighbor features while inferencing.

Experiments in multi-class emotional domains, ideally with more than two emotions, cause lots of biases in class predictions. On the other hand, if the classes are greatly separable in terms of features, feeding this characteristic into collective classification achieves significant accuracies.

It is possible to extend current study with some future works such as:

- Since all the user nodes are fully-labeled with their sentiments in this study,

same techniques can be evaluated on partially-labeled networked data where few labels of entities are known.

- Edge weights are assumed equally as 1 in all this thesis relationship data. For the networks that include weighted edges, different link mining techniques could be considered such as edge selection or handling heterogeneous links. Edge selection can propose techniques analogous to those used in traditional feature selection.

- Proposed and existing methods are applied into time-dependent emotional data that have changes in user emotions along the time.

- From the implementation view, the collective classification algorithms could be distributed in a cloud compliant engine like Apache Spark [1]. In this way, large-scale social networks can be analyzed as fast as possible so that algorithms running times are reduced.

[1] http://spark.apache.org

REFERENCES

[1] Fırat Akba, Alaettin Uçan, Ebru Akcapinar Sezer, and Hayri Sever. Assessment of feature selection metrics for sentiment analyses: Turkish movie reviews. In *8th European Conference on Data Mining 2014*, 2014.

[2] Ahmet Afsin Akın and Mehmet Dündar Akın. Zemberek, an open source nlp framework for turkic languages. *Structure*, 10:1–5, 2007.

[3] Cecilia Ovesdotter Alm, Dan Roth, and Richard Sproat. Emotions from text: machine learning for text-based emotion prediction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 579–586. Association for Computational Linguistics, 2005.

[4] Saima Aman and Stan Szpakowicz. Using roget's thesaurus for fine-grained emotion recognition. In *IJCNLP*, pages 312–318. Citeseer, 2008.

[5] Zeynep Boynukalin. Emotion analysis of turkish texts by using machine learning methods. Master's thesis, Middle East Technical University, 2012.

[6] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine, 1998. In *Proceedings of the Seventh World Wide Web Conference*, 2007.

[7] Soumen Chakrabarti, Byron Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. In *ACM SIGMOD Record*, volume 27, pages 307–318. ACM, 1998.

[8] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[9] Sinem Demirci. Emotion analysis on turkish tweets. Master's thesis, Middle East Technical University, 2014.

[10] Xiaowen Ding, Bing Liu, and Philip S Yu. A holistic lexicon-based approach to opinion mining. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, pages 231–240. ACM, 2008.

[11] Paul Ekman. Facial expressions. *Handbook of cognition and emotion*, 16:301–320, 1999.

[12] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1:12, 2009.

[13] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[14] Marti A. Hearst, Susan T Dumais, Edgar Osman, John Platt, and Bernhard Scholkopf. Support vector machines. *Intelligent Systems and their Applications, IEEE*, 13(4):18–28, 1998.

[15] Charles Kadushin. *Understanding social networks: Theories, concepts, and findings*. Oxford University Press, 2012.

[16] John T Kent. Information gain and a general measure of correlation. *Biometrika*, 70(1):163–173, 1983.

[17] Funda Kivran-Swaine and Mor Naaman. Network properties and social sharing of emotions in social awareness streams. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, pages 379–382. ACM, 2011.

[18] Zornitsa Kozareva, Borja Navarro, Sonia Vázquez, and Andrés Montoyo. Uazbsa: a headline emotion classification through web information. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 334–337. Association for Computational Linguistics, 2007.

[19] Qing Lu and Lise Getoor. Link-based classification. In *ICML*, volume 3, pages 496–503, 2003.

[20] Sofus A Macskassy and Foster Provost. Classification in networked data: A toolkit and a univariate case study. *The Journal of Machine Learning Research*, 8:935–983, 2007.

[21] Luke K McDowell and David W Aha. Labels or attributes?: rethinking the neighbors for collective classification in sparsely-labeled networks. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 847–852. ACM, 2013.

[22] Saif M Mohammad. # emotional tweets. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 246–255. Association for Computational Linguistics, 2012.

[23] Jennifer Neville and David Jensen. Iterative classification in relational data. In *Proc. AAAI-2000 Workshop on Learning Statistical Models from Relational Data*, pages 13–20, 2000.

[24] Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134, 2000.

[25] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.

[26] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[27] Claudia Perlich and Foster Provost. Aggregation-based feature invention and relational concept classes. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 167–176. ACM, 2003.

[28] Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[29] Juliano CB Rabelo, Ricardo BC Prudêncio, Flávia Barros, et al. Using link structure to infer opinions in social networks. In *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on*, pages 681–685. IEEE, 2012.

[30] Azriel Rosenfeld, Robert A Hummel, and Steven W Zucker. Scene labeling by relaxation operations. *Systems, Man and Cybernetics, IEEE Transactions on*, (6):420–433, 1976.

[31] S Shivashankar and Balaraman Ravindran. Multi grain sentiment analysis using collective classification. In *ECAI*, pages 823–828, 2010.

[32] Mansur Alp Tocoglu and Adil Alpkocak. Emotion extraction from turkish text. In *Network Intelligence Conference (ENIC), 2014 European*, pages 130–133. IEEE, 2014.

[33] Dilara Torunoğlu, Erhan Çakırman, Murat Can Ganiz, Selim Akyokuş, and M Zahid Gürbüz. Analysis of preprocessing methods on classification of turkish texts. In *Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on*, pages 112–117. IEEE, 2011.