Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

Ananth Mahadevan

Inferring Voting Networks in Online Elections

Master's Thesis
Espoo, March 14, 2020

DRAFT! — May 1, 2020 — DRAFT!

Supervisor:	Professor Aristides Gionis
Advisor:	Blank M.Sc. (Tech.)

Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

ABSTRACT OF MASTER'S THESIS

Author:	Ananth Mahadevan		
Title: Inferring Voting Networks in Online Elections			
Date:	March 14, 2020	**Pages:**	45
Major:	Computer Science	**Code:**	SCI3042
Supervisor:	Professor Aristides Gionis		
Advisor:	Blank M.Sc. (Tech.)		
abstract			
Keywords:	signed networks, balance, status, elections, Wikipedia, voting, graphs		
Language:	English		

Aalto-yliopisto
Perustieteiden korkeakoulu
Tieto-, tietoliikenne- ja informaatiotekniikan maisteriohjelma

DIPLOMITYÖN TIIVISTELMÄ

Tekijä:	Ananth Mahadevan		
Työn nimi: Äänestysverkkojen päätelmät online-vaaleissa			
Päiväys:	20. maaliskuuta 2020	**Sivumäärä:**	45
Pääaine:	Tietotekniikka	**Koodi:**	SCI3042
Valvoja:	Professori Aristides Gionis		
Ohjaaja:	Diplomi-insinööri Blank		
Finnish Abstract			
Asiasanat:	Finnish Keywords		
Kieli:	Englanti		

Aalto-universitetet
Högskolan för teknikvetenskaper
Magisterprogrammet i data-, kommunikations- och informationsteknik

SAMMANDRAG AV DIPLOMARBETET

Utfört av:	Ananth Mahadevan		
Arbetets namn: Avsluta omröstningsnätverk i onlineval			
Datum:	Den 20 mars 2020	**Sidantal:**	45
Huvudämne:	Datateknik	**Kod:**	SCI3042
Övervakare:	Professor Aristides Gionis		
Handledare:	Diplomingenjör Blank		
Swedish abstract			
Nyckelord:	Swedish Keywords		
Språk:	Engelska		

Acknowledgements

Espoo, March 14, 2020

Ananth Mahadevan

Contents

Chapter 1

Introduction

In recent years, researchers have become increasingly interested in understanding the behaviour of voters in social networks. Knowledge of the factors that motivate voters, for example, voting for bills in the United States Congress [27] or electing administrators in Wikipedia [9, 26, 32], is of great importance in selecting successful policies or candidates. Voting is a classic problem and has been studied in the fields of game theory and political science [29, 41, 49]. More recently, there is a focus on using information from the interaction network of voters to model their behaviour. This provides an insight into these interactions and effect of influence of other individuals on voters within a community.

Votes can be represented as a *signed* network with positive or negative links. Finding groups using clustering techniques [6, 12, 35] and predicting signed links [13, 33, 34] in these networks is well researched. These approaches provide understanding of the group dynamics at play and predict votes and in such a network. However, they do not consider the iterative and chronological nature of the voting that takes place in these networks. Moreover, in cases where research does focus on voter models, they rely on external features to build machine learning models that are task-specific and static [26, 27].

In this thesis, we propose a model that creates a local signed network consisting of the neighbours of the current voter and the votes already cast in the election in question. It will then predict the vote that when added to the network will comply the most with concepts of balance and status in signed networks. After all the votes are cast in a session, the model can be easily updated to improve quality and is, therefore, iterative and dynamic. The model is also flexible and can incorporate external features to build the local signed network of a voter. The results for Wikipedia administrator elections shows that our model outperforms machine learning based models and traditional signed link prediction solutions.

1.1 Thesis Outline

The rest of the thesis is organized in the following manner. Firstly, we discuss the background relating to signed graphs and hierarchy in directed networks in Chapter 2. Nexit, in Chapter 3 we describe the vote prediction problem and approaches to solving it. Chapter 4 provides a comprehensive view of Wikipedia and the election process for administrators. In Chapter 5 we explain the datasets used, construction of the model and evaluation criteria. After that, we report our findings in Chapter 6 and discuss their implications. Finally, we conclude the thesis and present future work in Chapter 7.

Chapter 2

Graph Theory

Change Chapter to Background and include signed link prediction theory here as well

In this chapter, we will provide the fundamentals of the graph theory concepts required to understand the rest of the thesis. In Section 2.1 we cover the basic definitions and terminologies used to describe different types of graphs. Then we define a signed graph and discuss its unique properties in Section 2.2. We outline the social theories of balance and status in signed networks in Sections 2.2.1 and 2.2.2. Lastly, we explain techniques of finding hierarchies in directed networks and the concept of agony in Section 2.3.

Add line for link pre-dic-tion

2.1 Preliminaries

In this section, we define the various types of graphs and their basic properties. The notation and terminologies used closely follow those used in Diestel [16]. Graphs are structures that describe relationships between entities. These entities are called *vertices* and entities related to one another are joined by edges. The terms graph, vertices and edges can be used interchangeably with *network*, *nodes* and *links* respectively.

Graphs can be classified broadly into two types based on whether the edges posses a direction. We now go on to define them in detail.

2.1.1 Undirected Graphs

An undirected graph is pair $G = (V, E)$, where V is the set of vertices and E is the set $E \subseteq \{(u, v) \mid u, v \in V\}$ of unordered pairs of vertices called edges. In this thesis we will deal only with *simple graphs*, i.e. no self loops,

$(u, v) \in V \times V,\ u \neq v$ and there is at most one edge between vertices u and v.

The number of the vertices in a graph is called the *order* of the graph and is denoted by $n = |G|$. In turn, the *size* of a graph is the number of edges denoted by $m = \|G\|$ or $m = |E|$. A vertex u is *adjacent* to v if they are the end points of an edge, $(u, v) \in E$. All the vertices adjacent to a vertex v is called the *neighbourhood* of v and is denoted by $N(v)$. The *degree* of a vertex v is the number of nodes adjacent to that vertex and is denoted by $d(v) = |N(v)|$.

The edges of an undirected graph can also have an associated value. This value can indicate the distance or similarity between a pair of vertices. These values are called *weights* and the corresponding graph is called a *weighted undirected graph*. Therefore, a weighted graph is defined as a triple $G = (V, E, w)$, where $w : E \to \mathbb{R}^+$ is a function that maps an edge e to a positive real weight $w(e)$. Now an *unweighted graph* is simply a weighted graph where the function w is defined as: if $e \in E$ then $w(e) = 1$ else $w(e) = 0$. The degree of a vertex v in a weighted graph is the sum of the weights to all the neighbours of v and is defined as $d(v) = \sum_{u \in N(v)} w((u, v))$. An example of a weighted undirected graph is shown in Figure 2.1.

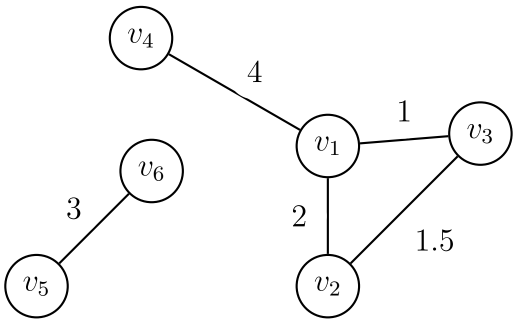


Figure 2.1: An example of a weighted undirected graph

2.1.2 Directed Graphs

The main distinction regarding a *directed graph* (or *digraph*) is that the edges are ordered pairs, i.e.$(u, v) \neq (v, u)$. Therefore, a directed graph has a similar definition: a pair $G = (V, E)$, where V is the set of vertices and E is the set of *ordered* pairs of vertices. Now given an edge $e = (u, v)$ we can define a source function $\mathrm{src} : E \to V$ such that $\mathrm{src}(e) = u$ and a destination function $d : E \to V$ where $\mathrm{dest}(e) = v$. These functions classify the vertices in an edge e as either the source or the destination. In this thesis, we deal only with *simple directed graphs*, i.e. no self-loops, and there can be at most one edge from u to v.

As the edges now have an inherent direction, we can define the *successors* and *predecessors* of a node v. A vertex u is called the *successor* of a node v if there exists a directed edge from v to u, therefore the set of successors for a vertex v can be defined as $S(v) = \{u \mid (v, u) \in E\}$. A *predecessor* of a node v is a vertex u such that there exists a directed edge from u to v, the set of predecessors for a vertex v can de defined as $P(v) = \{u \mid (u, v) \in E\}$. Now, a vertex u that is either a successor or a predecessor of a vertex v can be called a neighbour of the vertex v. Therefore, we define the *neighbourhood* of a vertex v as the set of vertices in the union of successors and predecessor, i.e. $N(v) = S(u) \cup P(v)$. This definition is also compatible with undirected graphs because if $(u, v) \in E$ then $(v, u) \in E$.

Directed graphs can also have values associated with each directed edge called a *weight*. A *weighted directed graph* can be defined as a triple $G = (V, E, w)$, where the weight function $w : E \to \mathbb{R}^+$ maps each edge e to a weight $w(e)$. The indegree of a vertex v is defined as the sum of the edge weights from the predecessors of v and is denoted as $\mathrm{d_{in}}(v) = \sum_{u \in P(v)} w((u, v))$. Similarly, the outdegree of a vertex v is defined as the sum of the edge weights to the successors of v and is denoted by $\mathrm{d_{out}}(v) = \sum_{u \in S(v)} w((v, u))$. Figure 2.2 shows an example of a weighted directed graph.

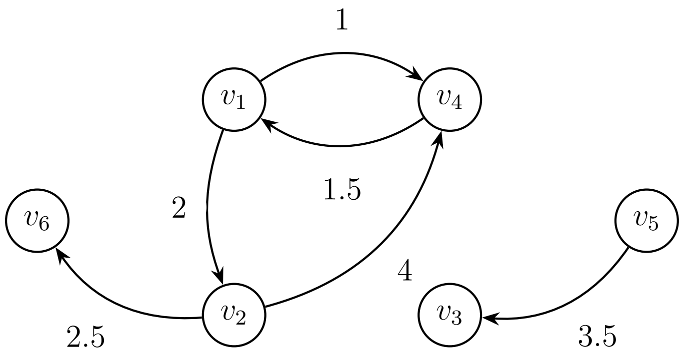


Figure 2.2: An example of a weighted directed graph

2.2 Signed Graphs

The simple weighted graphs we have defined so far only have non-negative edge weights that can represent similarity or closeness. In the 1950s, social psychologists found it desirable to express liking, disliking or indifference in social interactions. This was formalized by Harary [22] using graphs with weights $(-1, 0, 1)$. These graphs are therefore called *signed graphs*, where a negative edge weight can denote dissimilarity between a pair of vertices. In this thesis we will use notations and terms similar to Gallier [17], Kunegis et al. [31], Hou [24] and Zaslavsky [48].

A signed graph is a triple $G = (V, E, w)$, where V is the set of vertices, E is the set of pairs of vertices and the weight function $w : E \to \mathbb{R}$. The weight function now takes an edge e and maps it to a signed weight $w(e)$. We can partition the edges into positive and negative edges, $E = E^+ \cup E^-$, where $E^+ = \{e \mid w(e) > 0\}$ and $E^- = \{e \mid w(e) < 0\}$. Similar to Zaslavsky [48], we consider undirected signed graphs and directed signed graphs as distinct and separate entities. We can see some examples of signed graphs in Figure 2.3.

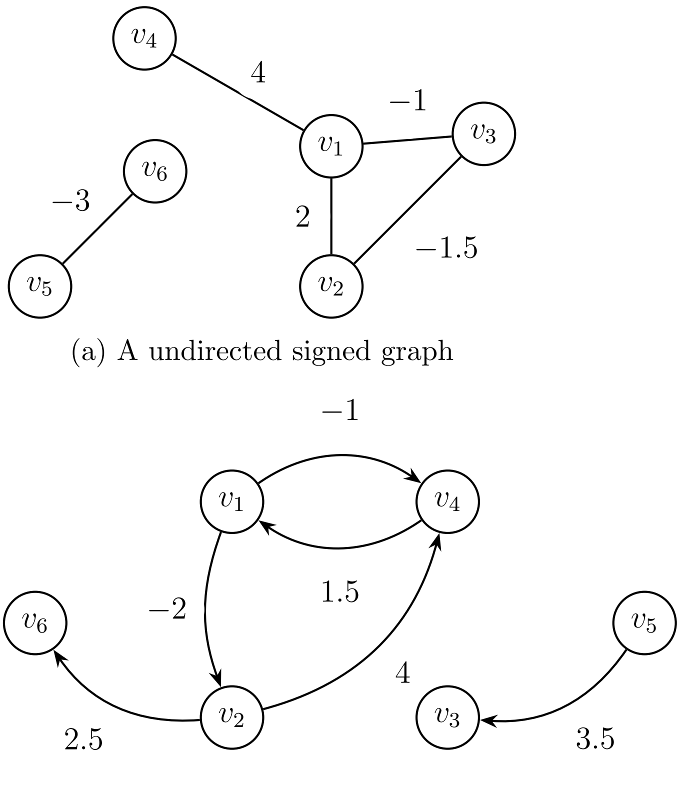


Figure 2.3: Examples of Signed Graphs

We can now proceed to define a few more terms. The degree of a vertex v is now the sum of the absolute edge weight of its neighbours, called the *signed degree* and is defined as

$$\overline{d}(v) = \sum_{u \in N(v)} |w((u, v))|$$

This can also be extended to *signed indegree* and *signed outdegree* denoted by $\overline{\mathrm{d_{in}}}(v)$ and $\overline{\mathrm{d_{out}}}(v)$ and defined as

$$\overline{\mathrm{d_{in}}}(v) = \sum_{u \in P(v)} |w((u, v))|$$

$$\overline{\mathrm{d_{out}}}(v) = \sum_{u \in S(v)} |w((v, u))|$$

We can create a $n \times n$ square weight matrix W, where each entry w_{ij} is defined as

$$w_{ij} = \begin{cases} w((v_i, v_j)) & \text{if } (v_i, v_j) \in E \\ 0 & \text{if } (v_i, v_j) \notin E \end{cases}$$

The signed degree matrix $\overline{D}$ is a diagonal matrix consisting of the signed verted degrees, $\overline{D} = \mathrm{diag}(\overline{d}(v_1), \ldots, d(v_n))$. We can now define the *signed Laplacian*, $\overline{L}$ as

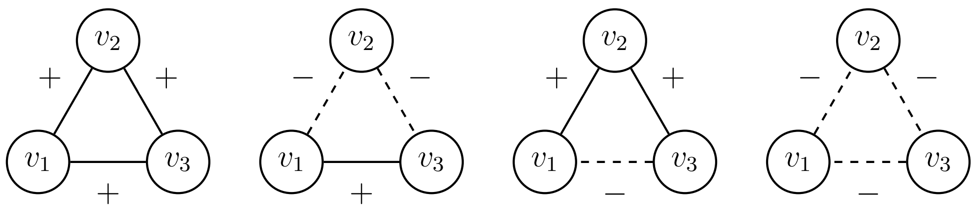
$$\overline{L} = \overline{D} - W$$

The signed Laplacian along with results from spectral analysis of signed graphs [24, 31], will be useful for balance theory of signed graphs.

2.2.1 Balance Theory

In the 1940s, Heider [23] proposed that when there are either *positive relations* (friendship, love, support) and *negative relations* (enmity, hate, oppose) in a group, the group tends towards *balance*. Balance is the concept that all members aim to maintain consistency in the relations they share with other members of the group. In a group of three members this can be seen as having three positive relations or one positive and two negative relations. In Figure 2.4 the triads B_1 and B_2 are *balanced* and mirror aphorisms such as "the friend of my friend is also a friend" and "the enemy of my enemy is a friend" respectively. Therefore, triads B_3 and B_4 can be called *unbalanced* or *imbalanced* that denotes the cognitive dissonance between the members of the triad. The dissonance can be understood by the counterintuitive nature of the phrase "the friend of my friend is my enemy" described by triad B_3.

Harary and Cartwright [10] generalized this notion of balance by using signed graphs. As these relations are typically symmetric, balance is usually defined for undirected signed graphs. This can been seen in Figure 2.4 where solid edges are positive and dashed edges are negative. Davies [14] also offers an alternate definition of *weak balance* where triads of type B_2 are considered to have lesser predictive utility as relationships of these type are less common in real-life social networks.



B_1 B_2 B_3 B_4

Figure 2.4: Triads in undirected signed graphs. B_1 and B_2 are *balanced* triads as they have even number of negative edges. B_3 and B_4 are *unbalanced* as they have odd number of negative edges.

The concept of balance can be generalized for an undirected signed graph $G = (V, E, w)$. The sign of a cycle C in the signed graph G is defined as the product of the edge weights $sgn(C) = \prod_{e \in C} w(e)$. A signed network G is then said to be balanced if and only if all the sign of all cycles in network are positive. Therefore, every cycle in G must have an even number of negative edges. This leads to result from Harary [22] that states that if a graph G is balanced, then there is a partition of the vertices $V = V_1 \cup V_2$ such that edges within the vertices of each set is positive and edges between the sets are negative. This means that we can have a bipartite graph when we delete the positive edges and negative edges span between the two sets of vertices. An example of a balanced signed graph is shown in Figure 2.5. Here the partition of the vertex set is $V_1 = \{v_1, v_3, v_4, v_7, v_8\}$ and $V_2 = \{v_2, v_5, v_6, v_9\}$.

The signed Laplacian matrix $\overline{L}$ of a signed graph G is always positive-semidefinite and $\overline{L}$ is positive-definite iff G is *unbalanced* [24, 31, 48]. If the smallest eigenvalue of a graph G is denoted by $\lambda_1(G)$, then G is balanced iff $\lambda_1(G) = 0$. Hou [24] provides bounds on the value of $\lambda_1(G)$ and Li et al. [36] show that $\lambda_1(G)$ can be used as a measure of how far the signed graph G is from being balanced.

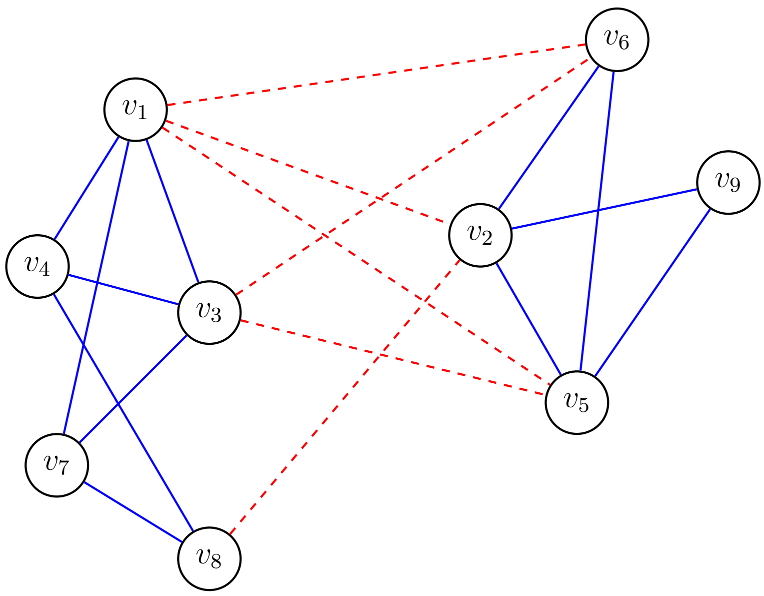


Figure 2.5: A balanced signed graph. Solid blue edges are positive and dashed red edges are negative. Every cycle in this graph contains an even number of negative edges.

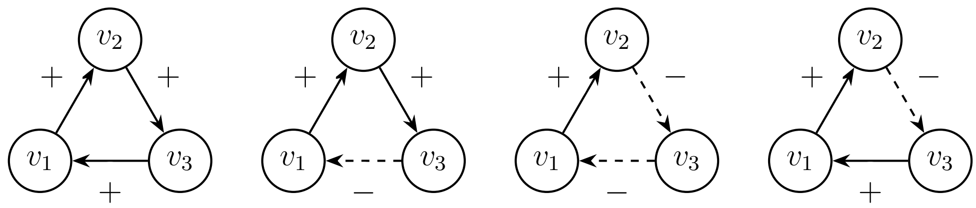
2.2.2 Status Theory

Guha et al. [20] mention implicitly that a signed edge from u to v can be interpreted in a asymmetric manner different from "friend" or "enemy". Leskovec et al. [33, 34] introduce the concept of *status* to contextualize directed singed edges. A positive edge $u \xrightarrow{+} v$ indicates that v has a higher status than u and a negative edge $u \xrightarrow{-} v$ means that v has a lower status than u. This concept of relative status can be propagated transitively along multiple steps which might lead to contradictions with balance theory [34].

Given three vertices v_1, v_2 and v_3, the presence of an edge $v_1 \xrightarrow{+} v_2$ indicates that v_1 thinks v_2 has higher status, the edge $v_2 \xrightarrow{+} v_3$ indicates that v_2 thinks v_3 has higher status. Now we wish to to close this triad with an edge from v_3 to v_1. Status theory would say that through transitivity v_1 has lower status than v_3, therefore the prediction is $v_3 \xrightarrow{-} v_1$. Whereas, in balance theory we would predict a positive edge $v_3 \xrightarrow{+} v_1$ to make the cycle have even number of negative edges. This example can be seen in triads S_1 and S_2 shown in Figure 2.6.

There are also cases when status theory is ambivalent to the edge that closes a triad. Consider the example when we have the edges $v_1 \xrightarrow{+} v_2$ and $v_2 \xrightarrow{-} v_3$. If we were to indicate the status of a vertex v using the function

$\sigma(v)$, then the edges describe the following : $\sigma(v_2) > s(v_1)$ and $\sigma(v_2) > \sigma(v_3)$. Therefore, we have no knowledge of the relative difference in status between the vertices v_1 and v_3. Hence, both edges $v_3 \xrightarrow{+} v_1$ ($\sigma(v_3) > \sigma(v_1)$) and $v_3 \xrightarrow{-} v_1$ ($\sigma(v_1) > \sigma(v_3)$) are equally valid for status theory. Balance theory on the other hand can only predict $v_3 \xrightarrow{-} v_1$ to keep the triad balanced. This case is shown in Figure 2.6 as triads S_3 and S_4.



S_1 S_2 S_3 S_4

Figure 2.6: Triads in directed signed graphs. Triads S_2, S_3 and S_4 are compliant with status theory. Only triads S_1 and S_3 are compliant with balance theory.

Each positive link inwards ($\mathrm{d_{in}}^+(v)$) and negative link outwards ($\mathrm{d_{out}}^-(v)$) increases status. Each positive link outwards ($\mathrm{d_{out}}^+(v)$) and negative link inwards ($\mathrm{d_{in}}^-(v)$) decreases status. Therefore, $\sigma(v) = \mathrm{d_{in}}^+(v) + \mathrm{d_{out}}^-(v) - \mathrm{d_{out}}^+(v) - \mathrm{d_{in}}^-(v)$ is a heuristic for the status of a node [33]. An interesting fact is that the edge $u \xrightarrow{-} v$ can be converted into positive edge in the opposite direction $u \xleftarrow{+} v$. This fact reduces the number of unique triads that can be formed using status theory and will be used in edge prediction tasks that will be discussed in coming chapters.

2.3 Hierarchy in directed networks

Hierarchies exist in all social structures, from the explicit levels found in nature such as the food chain or organizational structures in businesses to more implicit stratification that occurs on social media or online networks. A common method to represent such hierarchies is through a tree, for example, the chain of command in the military or within governments. Trees have well defined levels and a single person at the top. If we generalize this structural concept then we get a *Directed Acyclic Graph* (DAG) which represents a partial ordering set. DAGs have perfect hierarchy while structures such as cycles tend to have no hierarchy. Other directed graphs occur somewhere between these two extremes.

Gupte et al. [21] provide a method to discern the levels of stratification present in a given directed network when no prior information of the hierarchy exists. They define a measure of the hierarchy of a given directed network G as $h(G)$ along with a polynomial algorithm to find the largest hierarchy in that network. They define a concept of *social agony* which posits that agony is present when a person having a higher rank in network interacts with a person who has a lower rank. Therefore, if we define the rank of a node in graph G as the function $r : V \to \mathbb{N}$, then a directed edge $u \to v$ causes agony when $r(u) \geq r(v)$. The agony for the edge can be quantified as $\max(r(u) - r(v) + 1, 0)$. The agony of the graph G wrt to rank function r is defined as

$$A(G, r) = \sum_{(u,v) \in E} \max(r(u) - r(v) + 1, 0)$$

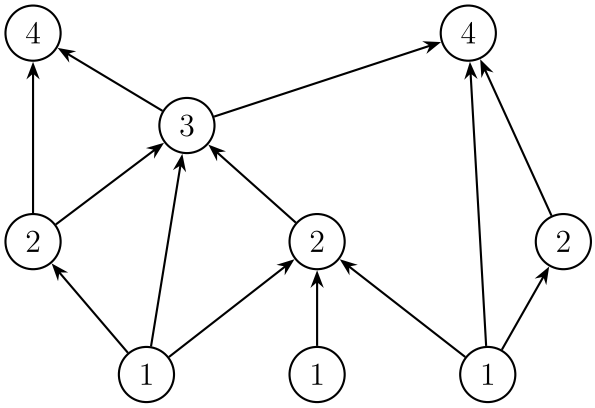
As nodes in a graph tend to minimize the overall agony, the agony of a network G is the smallest possible agony over all possible ranking for r, $A(G) = \min_{r \in \text{Rankings}} A(G, r)$. The hierarchy of a given network G, denoted by $h(G)$ can now be expressed in terms of the agony of the network, $h(G) = 1 - \frac{1}{m} A(G)$, where m is the number of edges. We can see examples of hierarchy in unweighted directed graph in Figure 2.7.

Therefore, finding a ranking of the nodes that minimizes the agony of the network gives us the optimal hierarchy present in that network. Gupte et al. [21] and Tatti [43] present a polynomial algorithm that can solve the dual of the agony minimization problem to obtain the optimal ranking r for unweighted graphs. Tatti [44] provides an alternate approach using a capacitated circulation solver that can handle weighted digraphs as well as additional cardinality constraints. These algorithms allows us to find the levels of hierarchy present in any given directed social network and analyse the interaction between members belonging to different strata in that community.

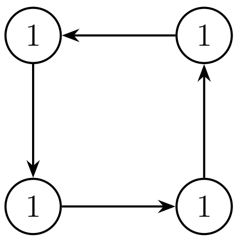
We will explain in future chapters how hierarchies in social network is intrinsically linked to the theory of status in directed signed networks. We show one can use the concept of agony of a directed signed graph G to quantify the violation to status theory and use it as a metric to predict an unknown signed edge.

2.4 Link and Sign Prediction

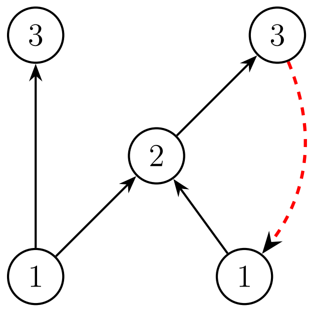
The *link prediction problem* is defined by Liben-Nowell and Kleinberg [37] as inferring possible future edges between vertices in a social network. The datasets used was split into training edges and testing edges which had a



(a) DAG has perfect hierarchy, $h(G) = 1$ and agony of each edge is 0



(b) Cycle has no hierarchy, $h(G) = 0$ and each edge has agony of 1



(c) Graph with some hierarchy $h(G) = \frac{2}{5}$. Red dashed edge has agony of 3 and solid black edges have 0 agony.

Figure 2.7: Examples of hierarchy in unweighted directed graphs. Numbers inside nodes indicate the rank of vertex.

common core set of vertices. They showed that topological features such as number of common neighbours and Katz's centrality index can be used in a unsupervised setting to accurately predict edges.

Leskovec et al. [33] extended the link prediction problem to signed networks. They also introduced the problem of *edge sign prediction*: predict the sign of a given edge (u, v) using the existing signed network G. A supervised model for the task is proposed that uses network features such as indegree and outdegree of a node along with *triad* features. The edge nodes u and v and a common neighbour w form a triad. The directed edge between (u, w) and (w, u) can be either forward or backwards and each edge is either positive or negative. Therefore there are 16 possible triad types for a given neighbour w. They analyse these triadic features from the trained model and show how they relate to balance and status theory. For the link prediction problem, the information from the negative edges in the signed network offers improvement in the overall accuracy of the model. This seminal paper inspired many more approaches to solving these problems for signed graphs.

Matrix factorization and latent space approaches facilitate link prediction and sign prediction tasks for multiple edges simultaneously in a signed network [1, 19, 25]. Supervised algorithms for both tasks can be improved using additional node features such as inverse square metric [2] or node rankings [40]. Models that use graph motifs [30, 38] generalize the concept of triadic feature for link and sign prediction. Chiang et al. generalize balance theory for longer cycles and use it as features to improve link prediction [13]. Tang et al. [42] discuss the importance of predicting negative links and highlight methods to overcome the inherent imbalance present in signed network datasets. Cesa et al. [11] and Chiang et al. [12] utilize clustering techniques to solve sign prediction and link prediction respectively. Shuang et al. [45] and Karimi et al. [27] create bespoke models incorporating user behaviour and political party affiliation respectively to predict the sign of edges present in the signed networks.

Chapter 3

Vote Prediction

In this chapter, we cover the main motivation behind predicting the vote of an individual voter and present the methods that can be used to solve this task. We discuss the contrast in perspectives that is present when predicting the result compared to predicting a vote in Section 3.1. Next, we explain how the problem of vote prediction is intrinsically linked to the tasks of edge and sign prediction in signed network in Section 3.2. In Section 3.3, we describe a supervised machine learning framework that can use graphs features from voting and non-voting data. Lastly, we present our novel approach of constructing a signed graph from neighbours of the current voter and previous votes and using balance and status theory to predict the vote in Section 3.4.

3.1 Result versus Vote Prediction

In this thesis, we are interested in the voting behaviour for a collective action. In such cases, members of a community come together as *voters* to decide on a particular *candidate* item during a *session*. In a government parliament the voters are the elected members of the parliament and the candidate is usually a bill or policy matter. When it is promotion within a political party or an online community such as Wikipedia, the members vote on a candidate who has been nominated for the position. In all these cases we have two levels of decisions being made. The first the individual decisions that a voter makes with regard to the candidate. The second is the final decision that the group arrives to after everyone has voted. We refer to task of predicting the former as *vote prediction* and the latter as *result prediction*.

Result prediction provides a macro level perspective of the incentives of a community. We can create models based on the characteristics of a candidate to predict the result of a collective action. This will lead to understanding on a communal level of what features are preferred and if there are voting blocks formed within the members based on the type of candidate. This translates to practical examples such as party level dynamics in a parliament, the topic of a bill or the credentials of a nominee [8, 46, 47].

On the other hand, when we focus on the vote prediction problem, we get a deeper understanding of the dynamics between voters and the candidate. In fields such as game theory, this is well studied using frameworks such as *strategic voting models* and *momentum* [3, 5, 39, 41, 49]. These models are more theoretical and are studied under synthetic conditions. Nevertheless, they still provide a foundation on which we can construct practical models that can utilize additional external features. One such popular approach is using textual information from bills, speeches and legislature to predict votes of politicians in parliament [7, 18]. The next important step is to represent the voting data as networks and leveraging network features to understand and predict voter behaviour.

3.2 Voting and Signed networks

Votes by nature express a positive or negative relationship between a *voter* and a *candidate*. Therefore, *signed graphs* provide an intuitive way to structurally represent the voting pattern of members in a community. These signed voting networks can be used to develop models to solve the task of vote prediction and analyse voter behaviour.

Correlation clustering and community detection of signed voting graphs can discover trends and vote blocks in communities [4, 6]. Analysing the networks using social theories of balance and status provides knowledge of voter behaviour and features for prediction methods [15, 35]. The vote prediction task can be broken down into two subtasks which are analogous to link and sign prediction in signed networks.

The first subtask is to predict who will vote next given a candidate c and a set of previous votes. This subtask is similar to link prediction in a signed network where we aim to predict possible future edges of the type (v, c). The complexity subtask depends on the format of voting that takes place. If there is a known voting order, such as in roll call in parliamentary proceedings or explicit timestamps then it is essentially solved. If the voting occurs simultaneously, the subtask can be reduced to predicting the possible subset of members who will vote in a given session. When the voting is iterative and there is no known underlying process of who votes next, then a separate model might be required to infer the voting sequence. This case can be combinatorially hard as we would need to find the correct ordering of votes in a session.

The second subtask is to predict how a voter v will vote for a candidate c given the previous votes in the session. This task translates into predicting the sign of an edge (v, c) in the current session. We call this subtask the *independent vote prediction* problem. It is independent in nature are we are only interested in the decision of the voter v assuming that we have complete prior knowledge of how the previous votes have been cast. This problem can be framed as a supervised learning task, using features of the interaction between the current voter v, previous voters U and the candidate c to predict the sign of the edge (v, c). We can utilize theories of balance and status in signed networks to create models similar to those by Karimi et al. [27] and Jankowski-Lorek et al. [26] to predict voter behaviour.

3.3 Linear Combination of Graphs

In this section we explain how the approaches outlined in Section 2.4 can be applied to solve the *independent vote prediction* problem. As discussed previously, the *edge sign prediction* task in signed network is analogous to vote prediction. The models proposed by Leskovec et al. [33] can be used to predict the sign of the edge (v, c). Voting in a community takes place across many sessions in a chronological manner. Therefore, we must partition the training and testing datasets to avoid data leakage. We propose the following framework to gather features using a linear combination of the voting history and several auxiliary graphs.

We denote the directed signed graph for the current voting session as $S = (V_S, E_S, w_S)$. The current voter in consideration is denoted by v and the candidate of the session is c. In this thesis, for all the models we assume that each session has a unique candidate. The votes that have been cast prior to the current voter exists as edges (u, c) in the session S and the set of prior voters is denoted as $U = \{u \mid (u, c) \in E_S\}$. The history $H = (V_H, E_H, w_H)$ is defined as the directed signed graph containing the votes from sessions chronologically prior to S. We also define a set of auxiliary graphs $A = \{G_1, G_2, \ldots, G_l\}$ based on external non-voting data. These auxiliary graphs can be either directed or undirected, weighted or unweighted, signed or unsigned. This is similar to the relational layer in *Multidimensional Social Networks* (MSN) described by Kazienko et al. [28] and Jankowski-Lorek et al. [26]. However, the auxiliary graphs capture different relations between a subset of the voting members which will be used to generate additional features for the vote prediction task.

Algorithm 1 describes how to generate a feature vector **a** from the auxiliary graphs A. The algorithm finds the intersection of the prior voters U and the neighbourhood of the current voter v in the graph G_i which we call the *voting neighbourhood*. Then the feature $\mathbf{a}_i$ is computed as the weighted sum of the voting neighbourhood plus the edge weight to the candidate (v, c). Figure 3.1 provides an example with three auxiliary graphs and two prior voters u_1 and u_2. The dashed red edges are the votes cast in the current session S by the prior voters. We see that in the example G_1 is a directed graph, G_2 is an undirected graph and G_3 is a signed directed graph. The current voter v has different relations to his voting neighbourhood in each auxiliary graph and therefore each feature is a different combination of edge weights.

In addition to the auxiliary feature vector $\mathbf{a}$, we can also create triad features based on the historical voting graph H. Similar to Leskovec et. al. [33] amd Karimi et al. [27] we can form a set of unique triads T. Then, for each node u in the common neighbourhood of N_{vc} we can count the triad formed from the three vertices. Algorithm 2 describes this procedure.

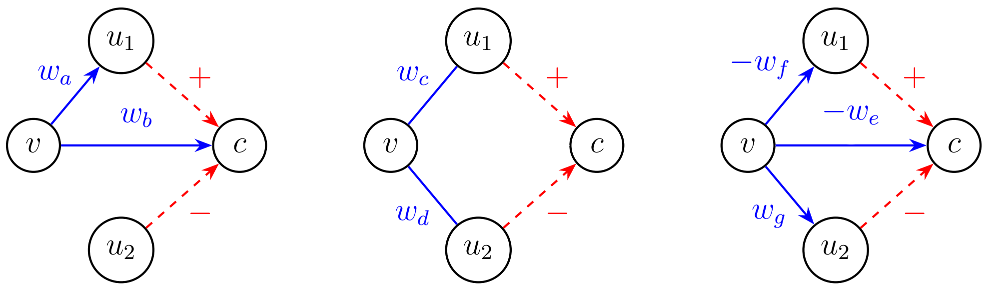
We can now create a feature matrix $\mathbf{X}$ for all the sessions in a given dataset. Each row is the concatenation of the auxiliary feature vector and the triadic feature vector $\mathbf{x} = [\mathbf{a}, \mathbf{t}]$. The target vector $\mathbf{y}$ is the vector of true votes. Now we train a linear machine learning model and use it to predict the votes in a test dataset.

Algorithm 1: Auxiliary feature vector for voter v

Input: Voter v, Candidate c, Set of auxiliary graphs A, Current voting session S and Prior voters U

Result: Auxiliary Feature vector $\mathbf{a}$

Initialize $\mathbf{a}$ of length $|A|$
foreach G_i *in* A **do**
    $Z = N_i(v) \cap U$ `// neighbours in` G_i `who have voted`
    $\mathbf{a}_i \leftarrow 0$
    foreach z *in* Z **do**
        `/* vote in current session multiplied by the edge weight in` G_i `*/`
        $\mathbf{a}_i \mathrel{+}= w_S((z, c)) \cdot w_i((v, z))$
    end
    $\mathbf{a}_i + = w_i((v, c))$ `// Add edge weight to candidate`
end
return a



G_1 $\quad$ G_2 $\quad$ G_3

$\mathbf{a}_1 = w_a + w_b$ $\quad$ $\mathbf{a}_2 = w_c - w_b$ $\quad$ $\mathbf{a}_3 = -(w_e + w_f + w_g)$

Figure 3.1: Example auxiliary features for v from combination of three graphs and two prior voters u_1 and u_2. Dashed red lines are prior votes in the session. Solid blue lines are edge weights in auxiliary graph. $\mathbf{a}_i$ is feature for voter v from auxiliary graph G_i

Algorithm 2: Triad feature vector for voter v

Input: Voter v, Candidate c, Set of unique triads T, Voting history graph H

Result: Triad Feature vector $\mathbf{t}$

$k \leftarrow |T|$

Initialize counters $cnt_1, \ldots, cnt_k$ to 0

$N_{vc} = N_H(v) \cap N_H(c)$ `// common neighbours in` H

foreach u *in* N_{vc} **do**

  Let $\triangle$ be the triad formed by vertices $\{v, u, c\}$

  Classify $\triangle$ as the jth triad in T

  Increment counter cnt_j

end

$\mathbf{t} \leftarrow [cnt_1, \ldots, cnt_k]$

return t

3.4 Local Signed Network

In this section, we present an unsupervised method that can be used iteratively in predicting the votes in a session. This method builds on top the concept of a *voting neighbourhood* introduced in the previous section and relies solely on the social theories of balance and status in signed networks to predict a vote. However, unlike the triads features used in the supervised method, we will utilize generic measures of the balance or status of a network and predict the vote that preserves these measures the best.

As defined in the previous section, the directed signed graph of the current session is $S = (V_S, E_S, w_S)$ and v is the voter, c is the candidate and U is the set of prior voters in the session. $H = (V_H, E_H, w_H)$ is the directed signed graph that contains the historical voting records prior to the current session. The first step is to construct a signed *relationship graph* $R = (V_R, E_R, w_R)$ from the history voting graph H. The edges of this graph capture simple signed relationship between the voters such as *agreement* or *concurrence* and can be constructed uniquely based on the given problem. Based on whether we use balance or status theory to predict the vote, the relationship graph is either unweighted or weighted respectively.

We now define the *Local Signed Network* (LSN) of a voter v as the intersection of the session and the relationship graph $LSN = S \cap R$. We assume that candidate c is present in the relationship graph so that we get all the prior vote edges (u_j, c) in the LSN. The *voting neighbourhood* that we described in the previous section is the set of vertices for the LSN i.e., $V_{LSN} = V_S \cap V_R$. There are three main types of edges present in the LSN. The first are the prior vote edges (u_j, c), from the prior voters to the candidate. The second are the relationship edges (v, u_j), from the voter to the prior voters. The third is the relationships between the prior voters (u_j, u_k). All these three types of edges can be seen in the undirected LSN shown in Figure 3.2c. We will now cover how to predict the edge (v, c) in the LSN using balance and status theory.

3.4.1 Prediction Based on Balance Theory

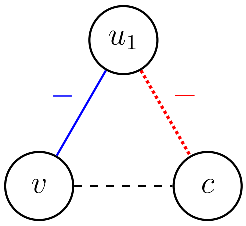
As described in Section 2.2.1, balance theory is applied to undirected signed graphs. Therefore, we construct a undirected signed relationship graph R using voting history. The signed relations the edges E_R in this graph describe should be symmetric in nature for e.g., the probability of agreements or disagreement between a pair of nodes u and v. Now, we create the LSN from the intersection of the session graph S and relationship graph R and keep the LSN an undirected graph by ignoring the direction of the voting edges (u_j, c).

Now, we turn to the task of predicting the sign of the edge (v, c) in the LSN. The voter v can cast either a positive or negative vote for the candidate c. This gives us two possibilities $w_{LSN}((v, c)) = +1$ or $w_{LSN}((v, c)) = -1$. We state that the voter aims to maintain the balance in the LSN. Therefore, we can predict the vote that when added as the edge (v, c) to the LSN results in a more balanced network. This indicates that we require a measure of the *imbalance* of a network. If $\overline{L}_{LSN}$ is the signed Laplacian of the LSN with eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \lambda_n$, then Li et al. [36] show that the smallest eigenvalue λ_1 is a measure of the imbalance of the LSN. Therefore, the larger λ_1 is, the more imbalanced the network is and if $\lambda_1 = 0$, then the network is perfectly balanced.

Combining these concepts we have Algorithm 3. When given a LSN and an edge (v, c) to predict, we first assume the vote is positive, add it to the LSN and compute the smallest eigenvalue denoted by λ_1^+. Next, we assume the vote is negative and add the edge to the LSN and the corresponding smallest eigenvalue is λ_1^-. Now, if $\lambda_1^+ < \lambda_1^-$, then we can predict that the vote is positive as the resulting LSN is more balanced. Similarly, if $\lambda_1^- < \lambda_1^+$, we predict a negative vote as it results in a more balanced network. This deterministic rule does not capture the gap between λ_1^+ and λ_1^-. Therefore, the ratio $r = \frac{\lambda_1^-}{\lambda_1^+}$ can be used to express the confidence in predicting the vote is positive. As λ_1^+ approaches 0 (or λ_1^- increases), r approaches ∞ and when λ_1^- approaches 0 (or λ_1^+ increases), r approaches 0. Also, when $\lambda_1^+ = \lambda_1^-$, $r = 1$ which indicates that we are equivocal between the vote being positive and negative. We can convert the ratio r into a measure of the probability that the given edge (v, c) is positive by defining $p = 1/(1 + e^{(1-r)})$. Therefore, the output of the model is a probability measure that can be compared similar to the output of a logistic regression model.

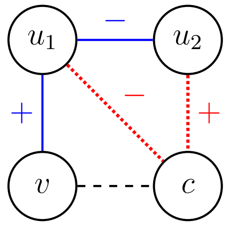
Figure 3.2 shows an example of how the we can iteratively predict votes using balance theory. The current voter at every iteration i is $v = u_{i+1}$. We start in the first iteration $i = 1$ with one prior voter u_1 who has voted negatively for the candidate c seen by the dotted red line in Figure 3.2a. The current voter v has a negative relation towards u_1 indicated by the solid blue line. Now, in this triad we know "v disagrees with u_1" and "u_1 disapproves of c". Using the intuition provided by balance theory, we can predict that the edge (v, c) is positive as it results in the triad being balanced. This result can be verified empirically by observing the values of smallest eigenvalue. We see that $\lambda_1^+ = 0$ and $\lambda_1^- = 1$ and therefore, the positive vote probability is $p = 1$. Now, in the next iteration, v becomes the prior voter u_2 and we add the edge (u_2, c) with the true vote (in this example we assume it was the same as the prediction made) and we get the next voter as the new v.

In the second iteration $i = 2$, the new voter v has a positive relation with the prior voter u_1 as seen in Figure 3.2b. By preserving the relation between u_1 and u_2 from the previous iteration there are larger cycles in the LSN. Similar to the previous iteration, we can observe the smallest eigenvalues and conclude that a negative vote leads to more balanced network and positive vote probability is now $p = 0$. Now, in the third iteration $i = 3$, the smallest eigenvalues in both cases are equal. Therefore, we are equivocal in the vote being positive or negative and the positive vote probability is $p = 0.5$.



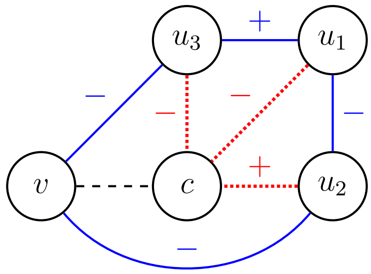
$\lambda_1^+ = 0, \lambda_1^- = 1$

(a) $i = 1$



$\lambda_1^+ = 0.58, \lambda_1^- = 0$

(b) $i = 2$



$\lambda_1^+ = 0.55, \lambda_1^- = 0.55$

(c) $i = 3$

Figure 3.2: Local Singed Network prediction based on balance theory. At every iteration i the dotted red lines are prior votes, solid blue lines are relationships based on voting records and the dashed black edge (v, c) is the edge whose sign is being predicted. λ_1^+ and λ_1^- correspond to the smallest eigenvalue of the signed Laplacian $\overline{L}_i$ based on whether $w((v, c)) = +1$ or $w((v, c)) = -1$.

3.4.2 Prediction Based on Status Theory

We mentioned in Section 2.2.2 that status theory is described for directed signed networks. The relationship graph R should, therefore, be be constructed from the history H as a directed signed network. The directed edges (u, v) should denote asymmetric relation between the nodes for e.g., the ratio of concurrence: a measure of times that u has voted after v in a session and agreed. The LSN created from the intersection of the session S and the relation is also a directed signed graph.

Algorithm 3: Predict positive vote probability using balance theory

Input: Voter v, Candidate c, Local Signed Network LSN
Result: Probablity of edge (v, c) being positive

$w_{LSN}((v, c)) \leftarrow +1$ `// Assume positive vote`
Compute signed Laplacian $\overline{L}_{+}$
$\lambda_1^{+} \leftarrow$ smallest eigenvalue of $\overline{L}_{+}$
$w_{LSN}((v, c)) \leftarrow -1$ `// Assume negative vote`
Compute signed Laplacian $\overline{L}_{-}$
$\lambda_1^{-} \leftarrow$ smallest eigenvalue of $\overline{L}_{-}$
$w_{LSN}((v, c)) \leftarrow 0$ `// Reset edge weight`
$r \leftarrow \lambda_1^{-}/\lambda_1^{+}$
$p \leftarrow 1/(1 + e^{(1-r)})$
return p

Similar to predicting the vote using balance theory, the vote is either positive or negative. Now, we state that the voter aims to maintain the status in the resulting LSN. Therefore, we predict the vote (v, c) that when added to the LSN best preserves status in the network. Therefore, we need a measure of how much a given network conforms to the theory of status. However, to the best of our knowledge, there are no existing methods to quantify and measure the extent that a network conforms to status theory. In this thesis, we present a quantitative definition of status theory in a network and also present a novel method of using the agony of a directed network to measure the compliance with status theory.

As we discussed in Section 2.2.2, signed edges can be interpreted as recognition of relative status. This means that a positive edge $u \xrightarrow{+} v$ indicates that v has a higher status than u, and a negative edge $u \xrightarrow{-} v$ indicates that v has lower status than u. Let us assume that there is an implicit status function $\sigma : V \rightarrow \mathbb{N}$ that maps each node in the network to a quantity that correlates with status. Therefore, the edges $u \xrightarrow{+} v$ and $u \xrightarrow{-} v$ indicate $\sigma(u) < \sigma(v)$ and $\sigma(u) > \sigma(v)$ respectively. This is how we predict the missing sign of an edge in a triad. Leskovec et al. [34] provide a way to compute this status function σ from the node degrees as seen in Section 2.2.2. However, this measure is still defined locally and can be used only to predict the sign of an edge. We still require a framework to quantify violations and measure how much the network complies with status theory. We define that an edge $e = (u, v)$ *violates* status theory if $w(e)\sigma(u) \nleq w(e)\sigma(v)$. Now, we can define when a network is perfectly complaint with status.

Definition 1. *Perfect Status Compliance A directed signed network* $G = (W, E, w)$ *with an implicit status function* $\sigma : V \to \mathbb{N}$ *is said to be perfectly status complaint if* $\forall e = (u, v) \in E$*,* $\operatorname{sgn}(w(e))\sigma(u) \leq \operatorname{sgn}(w(e))\sigma(v)$

Therefore, the number of edges that are in violation to status theory can be a rudimentary measure of the status compliance of a given network. However, in the case of most signed directed networks, we do not possess the implicit status function σ. Hence, even computing the number of edges that violate status theory is not possible. Therefore, if we can infer the status function σ from the structure of the signed network then we can measure the status compliance of that network.

We can now use the concept of *agony* described in Section 2.3 to find the optimal hierarchy of a given directed network. The notion of hierarchy is strongly related to status theory for signed networks. In a DAG, then there no cycles and we can find a status function σ such that there are no edges that violate status theory. Moreover, agony is a measure of the violation of edges with respect to a ranking function. Nevertheless, agony and hierarchy were primarily defined for unsigned networks. We can easily remedy this by using the fact that a negative edge $u \xrightarrow{-} v$ can be transformed into a positive edge $u \xleftarrow{+} v$. If G is a signed network then we denote the unsigned directed network obtained from the transformation described as G'.

Now, the agony of an edge (u, v) in G' given a status function (called rank function in Gupte et al. [21]) is $\max(\sigma(u) - \sigma(v) + 1, 0)$. Therefore, agony is a quantification of the status violation of an edge. The agony of the network with respect to a status function σ is defined as the sum of the agony of each edge in the network denoted by $A(G', \sigma)$. The agony of the network $A(G') = \min_\sigma(A(G', \sigma))$ is the smallest agony over all possible ranking of the nodes. In this way, the agony of the network is a more generalized measure of status compliance than just counting the number of violating edges. Therefore, we can use one of the algorithms mentioned in Section 2.3 to compute agony of G'. We call this value as the agony of the original signed network G and denote $\alpha = \alpha(G) = A(G')$. The complete process is outlined in Algorithm 4.Now, the agony of a signed network G is a measure of how far G is from being perfectly status complaint. Theorem 1 shows that when $\alpha = 0$ the network is perfectly status complaint.

Theorem 1. *Let* $G = (V, E, w)$ *be a directed signed graph. Then* $\alpha(G) = 0$ *if and only if* G *is perfectly status complaint.*

Proof. The transformed unsigned directed network is $G' = (V', E', w')$. The agony of a the directed network G^prime is 0 when the network has perfect hierarchy [21]. Therefore, there exists a status function σ such that the agony of all edges G' i.e., $\sigma(u) \leq \sigma(v), \forall(u, v) \in E'$. Therefore, there are no edges in G that violate status theory and G is perfectly status complaint. Hence proved. $\square$

Algorithm 4: Compute Agony for a directed signed network

Input: Directed siged graph $G = (V, E, w)$
Result: Signed Agony α of G
Initialize $G' = (V', E', w')$
$V' \leftarrow V$
foreach $e \in E$ **do**
  if $w(e) < 0$ **then**
    $e' \leftarrow (\text{dest}(e), \text{src}(e))$ `// Change direction of the edge`
    $E' \leftarrow E' \cup \{e'\}$
    $w'(e') \leftarrow -w(e)$ `// Make the weight positive`
  else
    $E' \leftarrow E' \cup \{e\}$
    $w'(e) \leftarrow w(e)$
  end
end
$\alpha \leftarrow Agony(G')$
return α

Now, we can compute the agony of the LSN and utilize it to predict the sign of the vote. We follow a process similar to predicting the sign using balance theorem. First, first assume the vote is positive and add the edge (v, c) to the LSN and compute the agony and call it as α^+. Next, we assume the vote is negative and add the edge and compute the agony of the LSN and call it α^-. If $\alpha^+ < \alpha^-$, then it means that the positive vote results in a LSN that has fewer violations of status and therefore, we can predict the vote is positive. Similarly, we predict a negative vote if $\alpha^- < \alpha^+$. Alternatively, we can also compute the probability of a positive vote as $p = 1/(1 + e^{(1-r)})$, where $r = \alpha^-/\alpha^+$. This process is detailed in Algorithm 5.

We see a directed LSN similar to the one in Figure 3.2a in Figure 3.3. The branches indicate the two possibilities for the vote edge (v, c). The left branch assumes that the vote is positive and adds it to the LSN. When we transform the negative edges in the LSN we get a cycle. As a cycle indicates that there is no hierarchy in the LSN the agony $\alpha^+ = 3$ reflects the fact that each edge violates status theory. The right branch assumes that the vote is negative and includes it in the LSN. After transformation, we see that all the edges comply with status theory and therefore, the agony $\alpha^- = 0$. Now, the corresponding positive vote probability is $p = 0$ indicating that we would predict a negative vote. Note this result is contradictory to balance theory where we predict a positive vote for the same LSN.

Algorithm 5: Predict positive vote probability using status theory

Input: Voter v, Candidate c, Local Signed Network LSN
Result: Probablity of edge (v, c) being positive

$w_{LSN}((v,c)) \leftarrow +1$ `// Assume positive vote`
$\alpha^+ \leftarrow SignedAgony(LSN)$
$w_{LSN}((v,c)) \leftarrow -1$ `// Assume negative vote`
$\alpha^- \leftarrow SignedAgony(LSN)$
$w_{LSN}((v,c)) \leftarrow 0$ `// Reset edge weight`
$r \leftarrow \alpha^-/\alpha^+$
$p \leftarrow 1/(1 + e^{(1-r)})$
return p

3.4.3 Iterative Prediction Model

Now, Algorithm 6 describes a model that can predict the votes in a session iteratively. We have the order of votes in the session denoted by O and the true votes function w^*. We create the session graph and initialize it with the candidate c and the first voter as seen in line 6. This is because we need at least one prior vote information to effectively predict a vote. In most settings, such as bills in a government parliament or promotion of a member in a community, there are always sponsors or nominators who propose the candidate in that session. Therefore, we are justified in starting the session with the candidate and the first vote cast in the session graph S. Next, for each subsequent voter v in the list, we add it the session and create the LSN. The *Predict* function in line 12 can be based on either balance (Algorithm 3) or status theory (Algorithm 5). Then, we add the true vote of voter v to the session G and move on to the next voter in the list. After all the votes are predicted in the session, we can update the relationship graph H with the data from the current session S. This can include operations such as adding nodes that were not present in R and updating the edge weights based on the votes cast in the current session.

In this process we can predict all the votes in all the sessions by starting with an empty relationship graph and updating it after every session. Therefore, the model can be compared to a "*batch*" machine learning model,

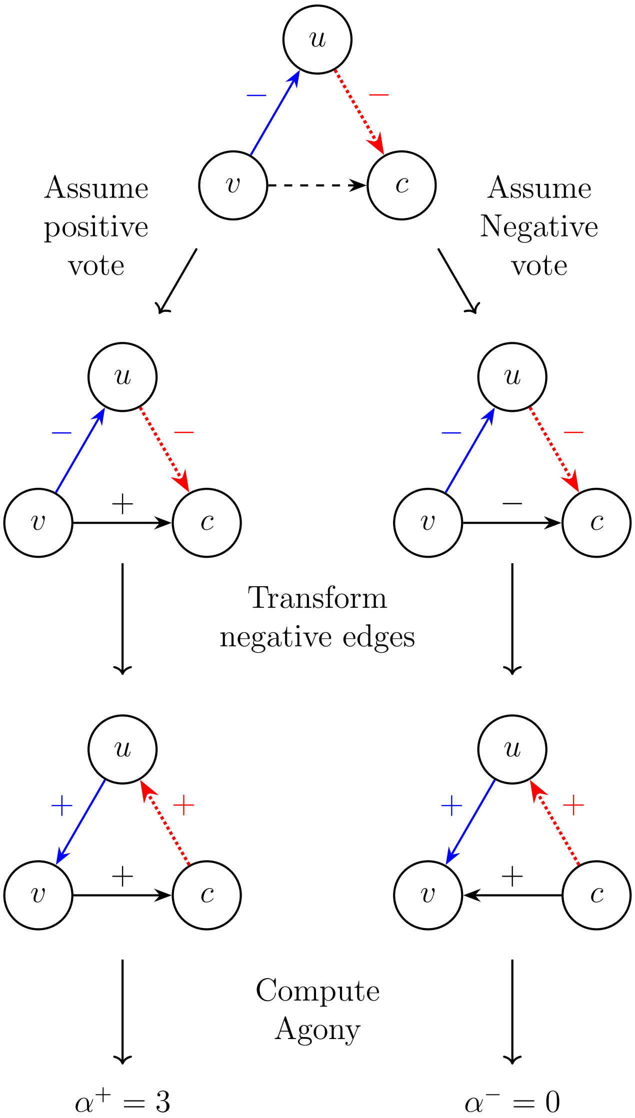


Figure 3.3: Example of LSN sign prediction using status theory.

where each batch is a voting session and the model parameter is the relationship graph R. In a batch, the model will predict votes based the paramters which is the information gathered from the previous sessions contained in the relationship graph R. After the batch is complete, the model updates its "parameters" or the graph with data from the current session. We can bootstrap the model from a complete *blank slate* where R is an empty graph and then iteratively predicts sessions and updates R. The model iteratively improves.

Algorithm 6: Iterative Prediction Model

Input: Candidate c, Relation graph $R = (V_R, E_R, w_R)$, Order of voters in current session O and true votes w^*

Result: Predictions for current session

$k \leftarrow |O|$
$u \leftarrow O[1]$ `// First voter`
$V_S \leftarrow \{c, u\}$ `// candidate and first voter`
$E_S \leftarrow \{(u, c)\}$ `// first vote`
$w_S((u, c)) \leftarrow w^*((u, c))$ `// Assign true vote`
Initialize session graph $S = \{V_S, E_S, w_S\}$
$predictions \leftarrow \emptyset$
for $i \leftarrow 2$ **to** k **do**
    $v \leftarrow O[i]$
    $V_S \leftarrow V_S \cup \{v\}$
    $LSN \leftarrow S \cap R$
    $p \leftarrow Predict(v, c, LSN)$
    $predictions \leftarrow predictions \cup p$
    $E_S \leftarrow E_S \cup \{(v, c)\}$
    $w_S((v, c)) \leftarrow w^*((v, c))$ `// Assign true vote`
end
$Update(R, S)$ `// Update Relations graph`
return $predictions$

Chapter 4

Wikipedia

In this section we provide an overview of how Wikipedia is structured, the hierarchy that exists withing editors. We then explain the election process of getting administrator rights in Wikipedia.

4.1 Structure and hierarchy in Wikipedia

4.2 Elections in Wikipedia

- Explain Editors and Administrators in Wikipedia
- Describe the Request for Administrator(RfA) process
- Discuss general trends and patters
- Mention research interest and possible current works?

Chapter 5

Experiments

In this section we first describe the datasets that will be used in building our vote prediction models. Then we discuss the various linear and graphical models that we consider and their implementations details. Lastly we define the metrics and other means of evaluating the models and the results.

5.1 Datasets

- Maybe a short description of existing SNAP datasets and their limitations
- The details of the *Wiki-RfA* data and the *User-Contribution* datasets

5.2 Graphs

- Discuss the process of extraction of the various graphs discussed in the previous sections
- **Agree Graphs and Follows Graph**, where we measure the degree to which one user agrees and follows another user in previous elections
- **Topic similarity** from the top 100 articles edited for each user and the pairwise Jaccard similarity
- **Talk and Interaction graphs**, measures communication between users on their respective user talk pages
- **Signed Graphs**, triad encoding and extracting the triad counts for each voter

5.3 Models

5.3.1 Linear Combination of Graphs

- Discuss the various linear models considered for Graph Combinations
  - Linear Regression
  - Support Vector Classifier
  - Extreme Gradient Boosting (XGBOOOST)
- Discuss how each graph contributes features and the problem is a linear classification problem

5.3.2 Iterative Mode

- Discuss the motivation behind an iterative model versus a static prediction model
- Describe how balance is derived from the Agree Graph in a local signed network
- Discuss how the Agree graph is updated in terms of Balance
- Describe how status is derived from the Follows graph in a local signed network
- Discuss how the Follows graph is updated after every election
- Describe how to make the predictions
  - Deterministic : just decide based on eigen value or agony as support or oppose
  - Probabilistic : provide a probability for predicting a support vote

5.4 Evaluation

- Discuss the issues with the imbalance in the datasets
- Illustrate the issues with pure measures of accuracy
- Define Precision, Recall and Macro F1 score
- Discuss ROC AUC and Precision Recall curves for probability based predictions

Chapter 6

Results and Discussion

In this section we will present the results of the models and discuss their implications.

6.1 Linear Combination of Graphs

- Present results for each linear classifier
- Discuss the different splits of the dataset to check for robustness and chronological consistency
- Show the feature importances and discuss their relevance
- Compare the raw accuracy versus the macro f1 scores
- Highlight the difficulty of predicting negative votes

6.2 Local Signed Network

- Present the Iterative Balance model results
- Discuss quality of predictions using evaluation metrics
- Mention the difference between deterministic and probabilistic prediction accuracies
- Explain the Iterative Status model results
- Discuss the issues with local model of status and the potential reasons for lower score and quality

6.3 Comparison

- Compare results from signed edge prediction and Iterative signed models
- Discuss Static Linear combination predictions versus Iterative signed predictions
- Discuss the assumptions used in the models and limitations

Chapter 7

Conclusions and Future Work

- Explain the quality of results with the election perspective
- Future work is to extend this to other election settings and investigate generality of this approach
- Possible future work in congressional voting data
- Can also tackle the other problem in information cascade theory of how to predict who is most likely to vote next
- This can lead to a complete model of election dynamics and could incorporate elements of game theory and network inference

Bibliography

[1] Agrawal, P., Garg, V. K., and Narayanam, R. Link label prediction in signed social networks. In *Twenty-Third International Joint Conference on Artificial Intelligence* (2013).

[2] Ahmadalinezhad, M., and Makrehchi, M. Sign prediction in signed social networks using inverse squared metric. In *International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation* (2018), pp. 220–227.

[3] Ali, S. N., and Kartik, N. A theory of momentum in sequential voting.

[4] Arinik, N., Figueiredo, R., and Labatut, V. Signed graph analysis for the interpretation of voting behavior, 2017.

[5] Banerjee, A. V. A simple model of herd behavior. *The quarterly journal of economics 107*, 3 (1992), 797–817.

[6] Brito, A. C. M., Silva, F. N., and Amancio, D. R. A complex network approach to political analysis: Application to the brazilian chamber of deputies. *PLOS ONE 15*, 3 (2020).

[7] Budhwar, A., Kuboi, T., Dekhtyar, A., and Khosmood, F. predicting the vote using legislative speech. In *Proceedings of the 19th Annual International Conference on Digital Government Research* (2018), p. 35.

[8] Burke, M., and Kraut, R. Mopping up: modeling wikipedia promotion decisions. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work* (2008), pp. 27–36.

[9] Cabunducan, G., Castillo, R., and Lee, J. B. Voting behavior analysis in the election of wikipedia admins. In *2011 International Conference on Advances in Social Networks Analysis and Mining* (2011), pp. 545–547.

[10] Cartwright, D., and Harary, F. Structural balance: a generalization of heider's theory. *Psychological Review 63*, 5 (1956), 277–293.

[11] Cesa-Bianchi, N., Gentile, C., Vitale, F., and Zappella, G. A correlation clustering approach to link classification in signed networks. In *Annual Conference on Learning Theory* (2012), vol. 23, Microtome, pp. 34–1.

[12] Chiang, K.-Y., Hsieh, C.-J., Natarajan, N., Dhillon, I. S., and Tewari, A. Prediction and clustering in signed networks: a local to global perspective. *Journal of Machine Learning Research 15*, 1 (2014), 1177–1213.

[13] Chiang, K.-Y., Natarajan, N., Tewari, A., and Dhillon, I. S. Exploiting longer cycles for link prediction in signed networks. In *Proceedings of the 20th ACM international conference on Information and knowledge management* (2011), pp. 1157–1162.

[14] Davis, J. A. Structural balance, mechanical solidarity, and interpersonal relations. *American Journal of Sociology 68*, 4 (1963), 444–462.

[15] Derr, T., and Tang, J. Congressional vote analysis using signed networks. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)* (2018), IEEE, pp. 1501–1502.

[16] Diestel, R. *Graph Theory.* 1997.

[17] Gallier, J. Spectral theory of unsigned and signed graphs. applications to graph clustering: a survey. *arXiv preprint arXiv:1601.04692* (2016).

[18] Gerrish, S., and Blei, D. M. Predicting legislative roll calls from text. In *Proceedings of the 28th International Conference on Machine Learning* (2011), pp. 489–496.

[19] Gu, S., Chen, L., Li, B., Liu, W., and Chen, B. Link prediction on signed social networks based on latent space mapping. *Applied Intelligence 49*, 2 (2019), 703–722.

[20] Guha, R., Kumar, R., Raghavan, P., and Tomkins, A. Propagation of trust and distrust. In *Proceedings of the 13th international conference on World Wide Web* (2004), pp. 403–412.

[21] Gupte, M., Shankar, P., Li, J., Muthukrishnan, S., and Iftode, L. Finding hierarchy in directed online social networks. In *Proceedings of the 20th international conference on World wide web* (2011), pp. 557–566.

[22] Harary, F. On the notion of balance of a signed graph. *Michigan Mathematical Journal 2*, 2 (1953), 143–146.

[23] Heider, F. Attitudes and cognitive organization. *The Journal of Psychology 21*, 1 (1946), 107–112.

[24] Hou, Y. P. Bounds for the least laplacian eigenvalue of a signed graph. *Acta Mathematica Sinica 21*, 4 (2005), 955–960.

[25] Hsieh, C.-J., Chiang, K.-Y., and Dhillon, I. S. Low rank modeling of signed networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (2012), pp. 507–515.

[26] Jankowski-Lorek, M., Ostrowski, L., Turek, P., and Wierzbicki, A. Modeling wikipedia admin elections using multidimensional behavioral social networks. *Social Network Analysis and Mining 3*, 4 (2013), 787–801.

[27] Karimi, H., Derr, T., Brookhouse, A., and Tang, J. Multifactor congressional vote prediction. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (2019), pp. 266–273.

[28] Kazienko, P., Musial, K., Kukla, E., Kajdanowicz, T., and Bródka, P. Multidimensional social network: model and analysis. In *International Conference on Computational Collective Intelligence* (2011), Springer, pp. 378–387.

[29] Kearns, M. J., Judd, J. S., Tan, J., and Wortman, J. Behavioral experiments on biased voting in networks. *Proceedings of the National Academy of Sciences of the United States of America 106*, 5 (2009), 1347–1352.

[30] Khodadadi, A., and Jalili, M. Sign prediction in social networks based on tendency rate of equivalent micro-structures. *Neurocomputing 257* (2017), 175–184.

[31] Kunegis, J., Schmidt, S., Lommatzsch, A., Lerner, J., Luca, E. W. D., and Albayrak, S. Spectral analysis of signed graphs for clustering, prediction and visualization. In *SDM* (2010), pp. 559–570.

[32] Lee, J. B., Cabunducan, G., Cabarle, F. G., Castillo, R., and Malinao, J. A. Uncovering the social dynamics of online elections. *Journal of Universal Computer Science 18* (2012), 487–505.

[33] Leskovec, J., Huttenlocher, D., and Kleinberg, J. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web* (2010), pp. 641–650.

[34] Leskovec, J., Huttenlocher, D., and Kleinberg, J. Signed networks in social media. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2010), pp. 1361–1370.

[35] Levorato, M., and Frota, Y. Brazilian congress structural balance analysis. *Journal of Interdisciplinary Methodologies and Issues in Sciences* (2016).

[36] Li, H. S., and Li, H. H. A note on the least (normalized) laplacian eighva; ue of signed graphs. *Tamkang Journal of Mathematics 47*, 3 (2016), 271–278.

[37] Liben-Nowell, D., and Kleinberg, J. The link-prediction problem for social networks. *Journal of the Association for Information Science and Technology 58*, 7 (2007), 1019–1031.

[38] Liu, S., Xiao, J., and Xu, X. Link prediction in signed social networks: from status theory to motif families. *IEEE Transactions on Network Science and Engineering* (2019), 1–1.

[39] Meir, R., Gal, K., and Tal, M. Strategic voting in the lab: compromise and leader bias behavior. *Autonomous Agents and Multi-Agent Systems 34*, 1 (2020), 31.

[40] Shahriari, M., and Jalili, M. Ranking nodes in signed social networks. *Social Network Analysis and Mining 4* (12 2014).

[41] Tal, M., Meir, R., and Gal, Y. K. A study of human behavior in online voting. *adaptive agents and multi-agents systems* (2015), 665–673.

[42] Tang, J., Chang, S., Aggarwal, C., and Liu, H. Negative link prediction in social media. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining* (New York, NY, USA, 2015), WSDM '15, Association for Computing Machinery, pp. 87–96.

[43] Tatti, N. Faster way to agony discovering hierarchies in directed graphs. In *ECMLPKDD'14 Proceedings of the 2014th European Conference on Machine Learning and Knowledge Discovery in Databases - Volume Part III* (2014), pp. 163–178.

[44] Tatti, N. Tiers for peers: a practical algorithm for discovering hierarchy in weighted networks. *Data Mining and Knowledge Discovery 31*, 3 (2017), 702–738.

[45] Yang, S.-H., Smola, A. J., Long, B., Zha, H., and Chang, Y. Friend or frenemy? predicting signed ties in social networks. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2012), SIGIR '12, Association for Computing Machinery, pp. 555–564.

[46] Yano, T., Smith, N. A., and Wilkerson, J. D. Textual predictors of bill survival in congressional committees. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2012), pp. 793–802.

[47] Yogatama, D., Heilman, M., O'Connor, B., Dyer, C., Routledge, B. R., and Smith, N. A. Predicting a scientific community's response to an article. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing* (Edinburgh, Scotland, UK., July 2011), Association for Computational Linguistics, pp. 594–604.

[48] Zaslavsky, T. Signed graphs. *Discrete Applied Mathematics 4*, 1 (1982), 47–74.

[49] Zou, J., Meir, R., and Parkes, D. Strategic voting behavior in doodle polls. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing* (2015), pp. 464–472.