Manipulating Elections by Selecting Issues

By

Jasper Lu

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

in

Computer Science

May, 2019

Nashville, Tennessee

Approved: Date:

____________________________ ____________________

Professor Yevgeniy Vorobeychik

____________________________ ____________________

Professor Xenofon Koutsoukos

Abstract

Constructive election control considers the problem of an adversary who seeks to sway the outcome of an electoral process in order to ensure that their favored candidate wins. In this thesis, we first consider the computational problem of constructive election control via issue selection. In this problem, a party decides which political issues to focus on to ensure victory for the favored candidate. We also consider a variation in which the goal is to maximize the number of voters supporting the favored candidate. We present strong negative results, showing, for example, that the latter problem is inapproximable for any constant factor. On the positive side, we show that when issues are binary, the problem becomes tractable in several cases, and admits a 2-approximation in the two-candidate case. Finally, we develop integer programming and heuristic methods for these problems.

We also consider the related problem of constructing ideological point estimates for a population when given the past voting history of that population. We develop a technique to generate point estimates in any number of dimensions using a simple neural network, and relate how the derived point estimates can be used in an instance of constructive election control via issue selection.

Dedicated to my parents, and to Dr. Yevgeniy Vorobeychik for giving me the opportunity to learn in his classes, even when I was just a sophomore.

Contents

Page

List of Tables

Table Page

List of Figures

Chapter 1

Introduction

When a politician decides to enter an election, they likely come in with a pre-developed stance on a number of political issues. However, although a candidate might have their own opinions on a large variety of different political issues, only a few key issues play a major role in determining the results of a given election season. Gallup, using data collected in every presidential election season from 2004 to 2020, has isolated six specific issues as important to voters in those elections: terrorism / national security, the economy, healthcare, the federal budget deficit, immigration, and taxes. The relative importance of those issues to each other changes with each election season [2].

When politicians and their campaigns promote some issues over others, they can influence the importance of different issues to the voters. Depending on what issues become most salient, or what issues voters care most about, in an election season, more voters may choose to align themselves with one candidate over the other. This thesis studies the scenario of a politician deciding which issues to promote in their platform. More specifically, given a good idea of what stances an electorate has on a number of different issues, and given the politician's own chosen platform, which issues should a politician choose to promote?

To illustrate this problem, take three political issues: universal healthcare, immigration, and gun control. Suppose that all voters want universe healthcare and environmental regulation, and a slight majority wishes to restrict immigration. Now, consider two candidates in an election: one who supports universal healthcare, environmental regulation, and immigration, and a second who is opposed to all three. Clearly, in an election in which the electorate cares equally about three issues, the former candidate would win in a landslide.

However, if one party is able to skew discourse *entirely* towards immigration, the second candidate may narrowly win.

This thesis sets forth a model of elections in which this problem can be studied computationally. More specifically, we deal with the problem of election control through manipulating issues. In the model we present, we assume that voters and candidates can be represented as points in a vector space, which we refer to as an ideological space. In this space, each index of the vector represents one's (voter's or candidate's) position on a political issue. Furthermore, the preference ranking of candidates by a voter is determined by the distance between the voter and all candidates in this ideological space.

The following chapter will go over some background and related work into election control and the spatial model of elections, a framework in which our research operates in. Chapter 3 will go in depth on the main focus of our research, putting forth a model of election control which focuses on manipulating elections by selecting issues for voters to focus on. Chapter 4 will propose a model which can generate point estimates of voters in an electorate given a history of their voting patterns. Such point estimates can then be studied in our model of election control.

Chapter 2

Background and Related Work

Our work here is related to two areas of research on social choice: the spatial theory of elections and election control.

2.1 Spatial Theory of Elections

In modern political discourse, we often speak of candidates and people in terms of a *political spectrum* – a continuum among which we identify a person's ideology [1] For example, given two candidates c_1 and c_2, we might say "c_1 is to the left of c_2, or c_2 is to the right of c_1 to mean that a candidate is more or less conservative than another. A natural extension of this is to think of candidates and people as points in a one-dimensional space, where the greater the value of the point, the more conservative that candidate is. In this space, given points for two candidates c_1 and c_2, where $c_1 \neq c_2$, we can say that $c_1 > c_2$ or $c_2 < c_1$ if c_2 is more conservative than c_1.

Although we tend to make generalizations to a one-dimensional ideological spectrum in political discourse, quantifying a person's ideology is much more complicated than that. To overcome this limitation, some political scientists have in the past attempted to quantify political ideology in a two-dimensional space [3] This is what led to the creation of the Nolan Chart, (figure 2.1) which quantifies political ideology along two axes, one representing a person's economic freedom "score" and the other a representing a person's personal freedom "score".

The Nolan Chart does a better job of quantifying ideologies than the previous left-right spectrum However, there is no reason we cannot use an arbitrary number of dimensions in

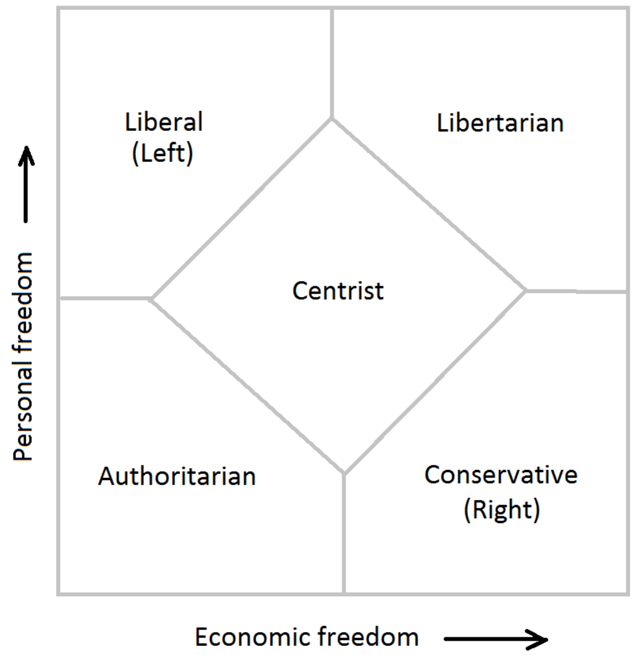

Figure 2.1: A simplified version of the Nolan Chart

our quantification, with each dimension representing an arbitrary type of "score." Our work is done primarily with this n-dimensional ideology space, in which a voter or candidate is represented by an n-dimensional real-valued vector.

Models such as the ones described above, first introduced by Hotelling [4] in 1929, are referred to colloquially as spatial models of elections. Since the introduction of the idea, extensive research has been done in the field. The following subsections will go over two key developments in the field of research.

Unfortunately, algorithmic work into spatial models of voting have been rather sparse, but there have been several recent studies focusing on social choice functions and distortion relative to a natural social choice function caused by common voting rules, such as plurality voting [5, 6, 7]

2.1.1 Median Voter Theorem

A major focus area of research in the spatial model of elections is that of a candidate choosing where to locate in a policy space [8, 9]. One key development in this field is the *Median Voter Theorem* (MVT), which dictates where a candidate should place themself in the one-dimensional version of the problem. Specifically, the median voter theorem states that "a majority rule voting system will select the outcome most preferred by the median voter [9]." So, given the choice, a candidate should place themself at the location of the median voter if they want to win an election.

However, although the median voter theorem was an important discovery in this model of elections, the assumptions the theorem relies on are rather unrealistic. The first assumption is that there is only one issue being voted on at a time. Or, reframed in the context of our specific model, we assume that voters and candidates can be represented in a single dimension. A second assumption that the MVT makes is that politicians care only about maximizing votes, and not about staying true to their beliefs. A third assumption is that voter's preferences are single-peaked. This means that voters have one alternative that they factor more than any other.

Research which aims to produce more realistic models of elections continue today. [10, 11].

2.1.2 Ideal Point Estimation

Given a spatial model, ideal point estimation deals with the problem of inferring from sample data a best guess of where a legislators and policies lie in a (generally) low-dimensional space. A large body of research into the field focuses on inferring ideal point estimates from congressional roll call data. These point estimates are then often used interest groups to produce "ratings" of legislators along policy dimensions, or to test legislative behavior theories.

One common technique for inferring point estimates, known as Nominal Three-Point Estimation (NOMINATE), uses a nonlinear logit model to infer one-dimensional and two-dimensional point estimates given a number of roll call voting records [12]. Another approach to point estimation in low-dimensional space makes use of a quadratic utility function, rather than the Gaussian one used in NOMINATE [13]. Some more recent research into ideal point estimation techniques uses neural networks and other deep learning techniques to infer ideal points for a variety of legislative uses [14].

2.2 Election Control

Election control research focuses on the problem of tampering with an election to either ensure that a candidate wins or loses an election. The spatial theory of elections aims to explain why voters vote the way they do by modeling an election system as sets of *voters* and *candidates* as positions in an n-dimensional policy space, in which voters vote for those candidates closest to them in Euclidean distance.

The computational problem of constructive election control, in which an adversary manipulates an election to ensure that a candidate wins was first studied by Bartholdi et al. [15], while Hemaspaandra et al. [16] initiated the study of destructive control. Much work since then has been done in election control under different voting systems, such as range voting [17], approval voting [18], and others [19, 20, 21, 22], as well as in bribery [23, 20, 24, 25].

2.3 Artificial Neural Networks

Artificial neural networks (ANNs) are a machine learning technique inspired in part by the way neural networks in our brain work. Usually used for classification problems, the technique trains a number of connected artificial neurons to recognize patterns in a given training dataset. We will provide a brief explanation on the way ANNs are trained and learn to classify data. For a more comprehensive overview, refer to online sources such as [26].

The basic unit of an artificial neural network is the neuron (also referred to as a node), which receives a number of inputs and then uses that input to spit out a single output. Let's say we have a neuron which takes n inputs. Then, we can let its input be a vector $x \in R^n$, where each x_i denotes the value of input i. In a basic neural network (also known as a feedforward nueral network), each neuron is generally associated with three attributes: a vector of w of weights in $\mathbb{R}^n$, a bias $b \in \mathbb{R}$, and a nonlinear activation function f which both takes and outputs a single number. One example of a common activation function is the sigmoid function, which takes a real-valued number and puts it somewhere between 0 and 1:

$$\sigma(x) = 1/(1+\exp(-x))$$

Given an input x, the output of the neuron will be $f(w^T x + b)$.

An artificial neural network generally consists of a number of hidden layers of neurons, where one layer of the network consists of neurons which all share the same input, but may have varying weights and biases. Figure 2.2 gives an example of such an artificial neural network with one hidden layer.

Assume we have a layer of an ANN with m neurons. Then, the layer will equivalently produce an output vector $y \in \mathbb{R}^m$. In an artificial neural network designed for classification between ℓ categories, the final layer of our ANN will consist of ℓ nodes. Most commonly, these nodes will share a softmax activation function, the purpose of which is to transform its input into an ℓ length output of values which sum up to 1. Each of these ℓ values corresponds to a probability that the input belongs to one of these categories.

Since the job of a neural network is to perform pattern recognition on a set of data, we must be given a set of training inputs X and corresponding outputs Y in order to train our neural network. One technique through which to train a neural network is through gradient

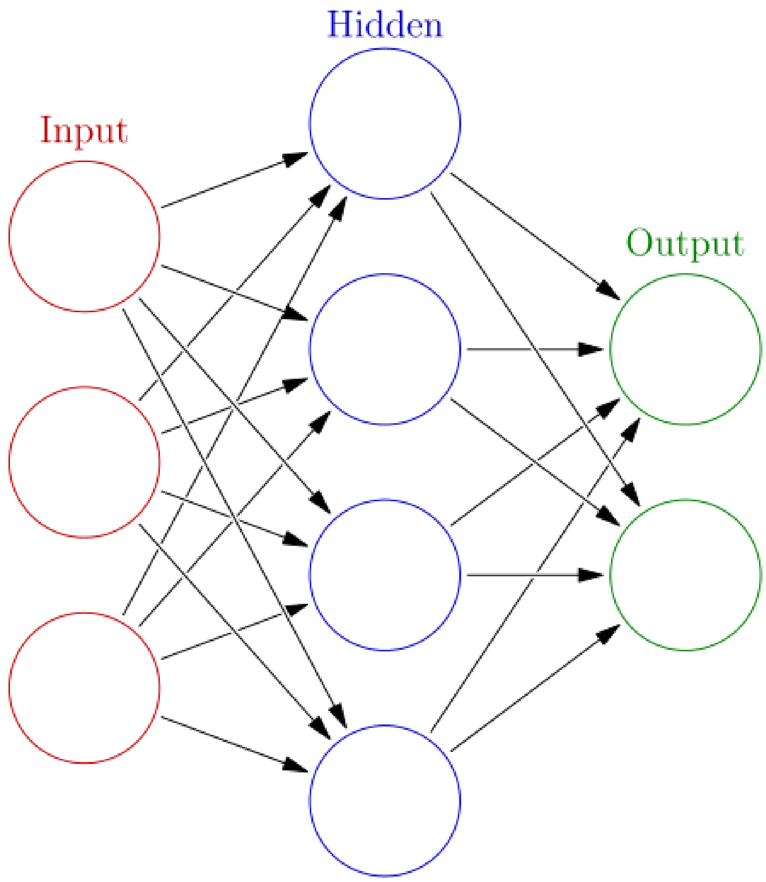

Figure 2.2: An example neural network diagram

descent with backpropagation. This technique has the neural network start out with random weights and biases. The network is run on all the training inputs, and an error is calculated between the outputs of our neural network and the true outputs. The network then adjusts the weights of the output layer to produce an output closer to the true outputs. Error is propagated backwards through the network via backpropagation.

We will not go over the specifics of backpropagation here, but the technique essentially passes the gradent of the error function from the final layer back through all of the previous hidden layers, so that we can adjust the weights and biases of each layer of the network.

2.3.1 Word Embeddings

Word embeddings are a way of modeling a language by mapping words or phrases from that vocabulary onto vectors of real numbers. They are useful because word embeddings can provide a way to encode phrases or words in a way that provides some kind of meaning to the computer, as opposed to just using a one-hot encoded vector to represent words. There are a variety of techniques through which word embeddings can be created. These include neural network models, dimensionality reduction techniques, or more probabilistic models. Perhaps the most popular technique used (and most relevant to our work here) is a technique developed by Google known as word2vec [27, 28].

Word2vec takes a corpus of text and uses it to derive word embeddings which can loosely represent relationships between words. Assume that we are given a set W of n words, indexed by W_i. We will represent each word W_i as a n-length one-hot encoded vector, where the i'th entry of the vector is a 1 but all other entries are 0. Then, given a corpus of text, we then create a training dataset as follows: take a sliding window of k words at a time, where k is an odd number. The middle word in these windows will become our training output. Then, the $\frac{k}{2}$ words on either side of that word will be our training input. Thus, from each window we can generate $k-1$ pieces of training data.

We train a shallow neural network (just one hidden layer) on this dataset. The network will take in a n-length vector as input, have m nodes in its hidden layer, and have n nodes in its output layer. Essentially, through this technique we will train a neural network to predict the likelihood of a word appearing, given the words surrounding it. However, we are not interested in the output of our neural network here, but rather the m-dimensional output of its hidden layer.

It has been shown that these word embeddings can provide very useful semantic information about words and their relationships to one another. Famously, researchers showed that the models produced via word2vec are good at determining association relationships between words, such as *king:queen::man:woman*, or *Rome:Italy::Beijing:China*.

Chapter 3

Election Control Through Issue Selection

Our work in controlling elections by manipulating issues uses a representation of candidates as voters and vectors in a multi-dimensional space. In our model, we refer to these vectors as a person's "belief vector." Each dimension of a belief vector is meant to represent that person's stance on a particular political issue. Within this framework, our study then investigates the problem of a candidate or third-party controlling an election by selecting issues to manipulate.

We study several related variations of this problem: the decision problem in which the interested party aspires to have a candidate of their choice win, and the optimization problem of maximizing the support (total number of votes) for a target candidate. All of our work here is done in the context of plurality elections, in which a candidate must receive more votes than any other candidate in order to win.

We obtain a series of strong negative results. First, we show that not only is the general problem of controlling elections through manipulating issues NP-Hard for both the decision problem and the variant aiming to maximize support, it is actually inapproximable for any constant factor for the latter variant. Moreover, the problem remains hard whether one breaks ties in favor of the target candidate, or not, and even when there is either a single voter, or two candidates. Second, we show that the problem remains hard if we restrict issues to be binary. On the other hand, we observe that under certain restrictions we can obtain positive results. For example, the problem is tractable if there is only a single voter (unlike in the general case), and maximizing support is 2-approximable when there are two candidates. Finally, we provide solution approaches for these problems based on integer linear programming, as well as a greedy heuristic.

3.1 The Model

Out model uses a multi-dimensional space to quantify political ideologies: consider a collection of ℓ political issues, and a space $X \subseteq \mathbb{R}^\ell$ of possible positions on the issues. Then, a vector $x \in X$ represents a vector of positions on all issues in this space, with x_k quantifying the position on (opinion about) issue k. In our setting, we have a collection of m candidates, $C = \{c_i\}_{i=1}^m$, and n voters, $V = \{v_j\}_{j=1}^n$, where each candidate i and voter j is characterized by a position vector (representing their respective positions on all ℓ issues), which we denote by c_i and v_j, respectively, with $c_i, v_j \in X$. We use c_{ik} (or v_{jk}) to denote the position of candidate i (voter j) on issue k, and we refer to the vector of a candidate's or voter's beliefs as their belief vector.

Denote by $[a:b]$ the interval of all natural numbers from a to b, and suppose that voters consider a nonempty subset of issues, $S \subseteq [1:n], S \neq \emptyset$, in deciding which candidate to vote for. This set S captures those issues which are *salient* to voters, for example, due to a focus on these during campaigning. We assume that a voter v_j will rank candidates in order of their relative agreement on issues, as captured by an l_p norm for integral $p \geq 1$ with respect to the set of issues S. Henceforth, we focus on plurality elections, so that a voter v_j would vote for a candidate c_i which minimizes $\|v_j^S - c_i^S\|_p$, where x^S denotes a restriction of x to issues in S.

3.1.1 Observations

Given a set of candidates C, as well as a set of voters V, we know that each voter v_j will vote for that candidate c_i which minimizes $\|v_j^S - c_i^S\|_p$. Or, put another way, a voter v_j will vote for the candidate they are closest to in ideological space. Pratt observes that, since voters vote for the candidate they are closest to in ideological space, each candidate c_i ends up commanding what he calls an *electoral zone* in that space [1]. Each voter captured by a specific electoral zone will end up voting for the candidate that that zone is owned by. These electoral zones actually end up being the Voronoi tessellations of the ideological space, given a set of candidates as the tesellating points, as shown by figure 3.1.

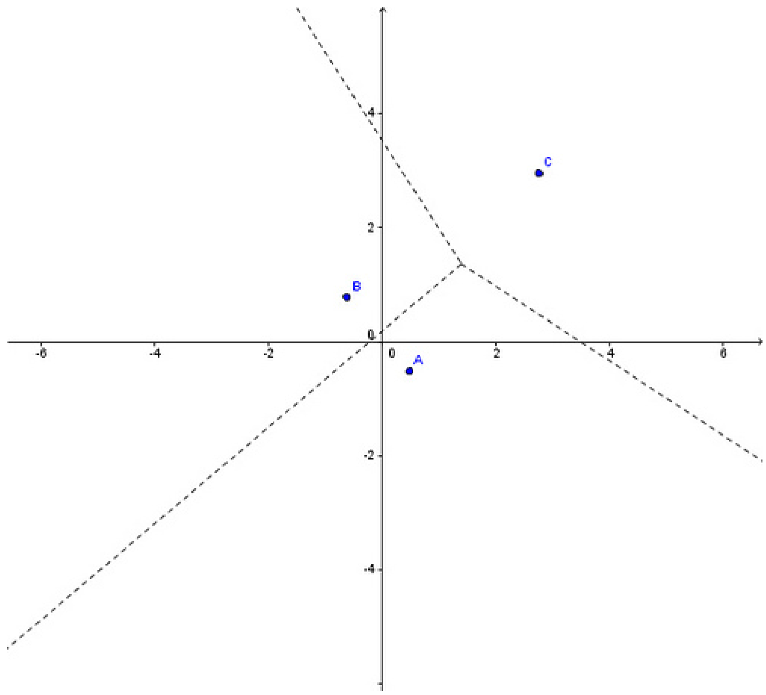

Figure 3.1: A three-candidate election in 2-dimensional space and each candidate's electoral zone (image taken from [1])

This would mean that, given a distribution or a set of voters V, one can predict the outcome of an election by creating a Voronoi tessellation of the space, and then counting the number of voters in each electoral zone. As a callback to the Median Voter Theorem, one can imagine a problem in which a candidate tries to find the point in n-dimensional space which gets them a plurality of voters in that space.

3.2 Problem Definition

We consider two *constructive control* problems within this framework: control through issue selection (ISSUE SELECTION CONTROL (ISC)), and maximizing support (MAX SUPPORT), which are defined formally as follows:

Definition 3.2.1 (ISSUE SELECTION CONTROL (ISC))**.** *Given a set of candidates C, voters V, and ℓ issues, is there a nonempty subset of issues $S \subseteq [1 : \ell]$ such that a target candidate c_1 wins the plurality election?*

Definition 3.2.2 (MAX SUPPORT)**.** *Given a set of candidates C, voters V, and ℓ issues, find a nonempty subset of issues $S \subseteq [1 : \ell]$ which maximizes the number of voters who vote for a target candidate c_1.*

For both problems, we must define a rule by which to break ties. We consider both the best-case of undecided voters choosing the target candidate c_1, and the worst-case of undecided voters choosing another candidate. We use the same tie-breaking rule when several candidates are tied.

3.3 Real-Valued Issues

We begin our study of election control by analyzing its algorithmic hardness when issue positions are unrestricted, i.e., $X = \mathbb{R}^\ell$. We show that the problem is computationally intractable, even for a single voter or with only two candidates. However, the problem is tractable when the number of issues is bounded by a constant.

3.3.1 Issue Selection with a Single Voter

Consider election control through issue selection with only a single voter, v, which we term SINGLE-VOTER (SVIS). We start by assuming that ties are broken in candidate c_1's favor (best-case tie breaking). Note that in this setting, and MAX SUPPORT are essentially equivalent: in either case, we ask whether there exists a nonempty subset of issues $S \subseteq [1:\ell]$ such that when restricted to these issues, candidate c_1 wins the voter v (with a maximum support of 1 if c_1 wins, and 0 if it loses). Equivalently, we ask if there exists a nonempty subset S such that

$$\sum_{k\in S} |c_{1k} - v_k|^p \leq \sum_{k\in S} |c_{ik} - v_k|^p \quad \forall i \in [2:m] \tag{3.1}$$

where v_k is the sole voter's position on issue k. Observe that condition (3.1) holds if and only if

$$\sum_{k\in S} |c_{ik} - v_k|^p - |c_{1k} - v_k|^p \geq 0 \quad \forall i \in [2:m]$$

Thus, setting the entries of an auxiliary $(m-1)\times\ell$ matrix M

$$M_{i-1,k} = |c_{ik} - v_k|^p - |c_{1k} - v_k|^p, i \in [2:m], k \in [1:\ell] \tag{3.2}$$

we can equivalently ask whether there exists a nonempty subset S of the columns of M such that the restriction of M to these has nonnegative row sums. We will refer to such a restriction of an election as "highlighting" a set of issues.

Theorem 3.3.1. SVIS *with best-case tie breaking is NP-complete for any l_p norm.*

Proof. First observe that SVIS is in NP. Indeed, given an instance of SVIS and a proposed subset S, it is trivial to verify whether S satisfies condition (3.1) in polynomial time.

We now show that SVIS is NP-hard via reduction from 0-1 INTEGER LINEAR PROGRAMMING, which is well-known to be NP-complete. In this problem, we are given a matrix $A \in \mathbb{Z}^{m\times\ell}$ and a vector $b \in \mathbb{Z}^{\ell}$, and we ask if there exists a vector $x \in \{0,1\}^{\ell}$ such that $Ax \geq b$ componentwise.

Given an arbitrary instance (A,b) of 0-1 INTEGER LINEAR PROGRAMMING (ILP), we construct an $(m+1)\times(\ell+1)$ matrix M as follows:

$$
\begin{aligned}
M_{i,k} &\coloneqq A_{i,k} && i=1,\dots,\ell && k=1,\dots,\ell\\
M_{i,\ell+1} &\coloneqq -b_i && i=1,\dots,m &&\\
M_{m+1,k} &\coloneqq -\frac{1}{\ell+1} && && k=1,\dots,\ell\\
M_{m+1,\ell+1} &\coloneqq 1. && &&
\end{aligned}
$$

This construction is motivated by the observation that choosing a subset S of columns of M so that c_1 wins the election is analogous to choosing the positions of ones in a vector x that satisfies $Ax \geq b$. Each row of M corresponds to a candidate belief vector with the constraint vector b included as an added issue. We force this issue to be considered by creating a dummy candidate whose beliefs coincide with c_1 on all but that issue.

We now construct an instance of SVIS by setting the voter belief vector v to be the zero vector and constructing a sequence of candidate belief vectors $C=\{c_i\}_{i=2}^{m}$ from M.

$$
\begin{aligned}
c_{1k} &\coloneqq \sqrt[p]{\left|\min_i M_{ik}\right|} && k\in[1:\ell+1]\\
c_{i+1,k} &\coloneqq \sqrt[p]{M_{ik}+c_{1k}^{p}} && i\in[1:m+1],\, k\in[1:\ell+1]
\end{aligned}
$$

We do this because we want to arrange that $M_{ik}=|c_{ik}|^p-|c_{1k}|^p$, using positive values of c_{ik} for simplicity. It is then straightforward to see that the original instance of 0-1 INTEGER LINEAR PROGRAMMING is satisfiable if and only if our constructed instance of SVIS is satisfiable, by constructing a 0-1 vector x with ones at precisely the indices in $S\setminus\{\ell+1\}$, or vice versa. □

Theorem 3.3.2. *The worst-case version of* SVIS *is at least as hard as the best-case version of* SVIS.

Proof Sketch. Consider an $m\times\ell$ matrix M representing an arbitrary instance of the best-case version of SVIS and define

$$\varepsilon = \min_{i\in[1:m],k\in[1:\ell]} \frac{1}{2}\left|\sum_{k'\in R(k)} M_{i,k'}\right|,$$

where the set $R(k) = \{\binom{r}{k}, r \in [1..\ell]\}$. We can create a new $(m+2)\times(\ell+1)$ matrix M' as follows:

$$\begin{aligned}
M'_{i,k} &\coloneqq M_{i,k} && i = 1,\dots,m && k = 1,\dots,\ell \\
M'_{m+1,k} &\coloneqq 0 && && k = 1,\dots,\ell \\
M'_{i,k+1} &\coloneqq \frac{\varepsilon}{2} && i = 1,\dots,m+1 && \\
M'_{m+2,k} &\coloneqq \varepsilon && && k = 1,\dots,\ell \\
M'_{m+2,\ell+1} &\coloneqq -\frac{\varepsilon}{2}. && &&
\end{aligned}$$

Recall that in the worst-case version of SVIS, a voter will default to other candidates in cases of a tie. So, we are forced to include issue $\ell+1$ in S in order to win against candidate $m+1$. Once we include issue $\ell+1$, we bias the voter towards the target candidate and against each other candidate by a small amount. Because of our choice of ε, this bias will only affect the election in instances where the candidates are tied. However, we still have to include at least one other issue from $[1:\ell]$ to win against candidate $m+2$.

This construction then turns into the best-case version of SVIS once we begin to consider combinations of issues from $[1:\ell]$ with issue $m+1$. □

3.3.2 Issue Selection with Two Candidates

While issue selection is hard even with a single voter, we now ask whether it remains hard if we have only two candidates. We term the resulting restricted problem TWO-CANDIDATE (TCIS). We show that both of the considered problem variants remain NP-hard. Furthermore, MAX SUPPORT is actually inapproximable to any constant factor even in this restricted setting.

Theorem 3.3.3. TCIS *with best-case tie breaking is NP-complete.*

Proof. First, observe that TCIS is in NP because, given a set S of issues to highlight, we can easily check if c_1 wins the election in polynomial time. We use a reduction from 0-1 INTEGER LINEAR PROGRAMMING to prove it's NP-Hard.

Next, consider the issue selection problem with two candidates and a set of voters V. Note that we successfully control the election iff the following condition holds for at least half of the voters v_j (remember that ties are broken in c_1's favor):

$$\sum_{k \in S} \left|c_{1k} - v_{jk}\right|^p \leq \sum_{k \in S} \left|c_{2k} - v_{jk}\right|^p \tag{3.3}$$

We now construct a matrix M with entries

$$M_{j,k} = \left|c_{2k} - v_{jk}\right|^p - \left|c_{1k} - v_{jk}\right|^p, j \in [1:n], k \in [1:\ell]. \tag{3.4}$$

We can equivalently ask for a nonempty subset S of columns of M such that the restriction of M to those columns maximizes the number of indices j s.t. $\sum_{k \in S} M_{jk} \geq 0$.

Let A be our ILP matrix, and b - the ILP constraints. Then, we can reduce ILP to TCIS by creating the following $(2n+1) \times (\ell+1)$ matrix M:

$$\begin{aligned}
M_{j,k} &:= A_{j,k} && j \in [1:n] && k \in [1:\ell] \\
M_{j,k} &:= -1 && j \in [n+1:2n+1] && k \in [1:\ell] \\
M_{j,\ell+1} &:= -b_j && j \in [1:n] \\
M_{j,\ell+1} &:= 0 && j \in [n+1:2n] \\
M_{2n+1,\ell+1} &:= \ell+1
\end{aligned}$$

As in our reduction of SVIS, we represent the constraint vector b as an issue that must be put in S in order for S to win. We also create n dummy voters with all negative entries. This will force us to look for assignments of S that satisfy all rows that correspond to constraints of ILP. If c_1 can win the given election, we return yes for ILP, and no if c_1 cannot win.

Finally, we show that for any M we can derive voter preferences consistent with it. Since definition of M is independent for different issues k, it will suffice to do this for an arbitrary issue k (kth column of M, which we denote by M_k). Consequently, consider a column M^k, and define $\bar{M}_k = \max_j |M_{j,k}|$ (the value of M_k with the largest magnitude). Define $c_{1k} = 0$ and $c_{2k} = \bar{M}_k^{1/p}$. Additionally, define a function $f(z) = |c_2 - z|^p - |c_1 - z|^p$ for $z \in [0, c_2]$. Clearly, this function is continuous, and $f(0) = \bar{M}_k$ while $f(c_2) = -\bar{M}_k$. Then by the intermediate value theorem, for any M_{jk}, we can find a v_{jk} such that $f(v_{jk}) = M_{jk}$. Repeating the process for each issue k gives us the construction. □

Next, we turn to the MAX SUPPORT version of the issue selection problem with two candidates; we term this TWO-CANDIDATE MAX SUPPORT (TCMS). We show that not only is it NP-hard, it is inapproximable.

Theorem 3.3.4. TCMS *with best-case tie breaking is NP-hard for any l_p norm. Moreover, it cannot be approximated to any constant factor unless $P = NP$.*

Proof. We can now show that TCMS is NP-hard by restricting ℓ to 2 and reducing from MAXIMUM INDEPENDENT SET (MIS). Given an undirected graph $G = (V, E)$ on $|V|$ vertices, MIS asks to select a maximal subset of vertices $S \subseteq V$ so that S is an independent set (i.e., no pair of vertices in S is connected by an edge).

Given any instance of MIS, we can represent that instance as an instance of TCMS by first creating a $|V| \times |V|$ matrix with every value along the diagonal equal to $|V| - 1$. For every pair of vertices u, v, set $M_{u,v} = M_{v,u} := -|V|$ if u and v are connected in G, and -1 otherwise. Now, if we were to select an issue corresponding to vertex u with neighbor v, then we cannot hope to select any other subset of issues such that row v sums to greater than or equal to 0. Thus, the action of selecting columns of M to include in S corresponds to selecting vertices of G to be in our independent set, and maximizing the number of rows in this manner corresponds to finding a maximum independent set.

To complete the reduction, what remains to prove is that we can derive voter belief and candidate belief vectors for any M constructed in this manner.

Lemma 3.3.5. *For any set of values in matrix F, we can derive candidate vectors C and V for* FIXED-CANDIDATE ISSUE SELECTION *such that they correspond with F. (note: fix up notation in the future. Each v from hereon corresponds with voter belief vectors)*

Proof. We show the appropriate assignments for an instance of FIXED-CANDIDATE ISSUE SELECTION in which $p = 1$, and then prove that the technique can be generalized for all p.

Let $p = 1$. Then, construct our set of candidates C and voters V as follows:

$$x_j := \max\{\left|\min_i F_{i,j}\right|, \left|\max_i F_{i,j}\right|\} \tag{3.5}$$

$$c_{1,j} := -x_j \tag{3.6}$$

$$c_{2,j} := x_j \tag{3.7}$$

$$v_{i,j} := \frac{F_{i,j}}{2} \tag{3.8}$$

We do this because we want to arrange that $F_{i,j} := \left|c_{1,j} - v_{i,j}\right| - \left|c_{2j} - v_{i,j}\right|$. If $c_{1,j}$ and $c_{2,j}$ are sufficiently small and large enough, respectively, for each issue j, then we can represent any number in between the two values easily.

The above assignments are specifically for an FIXED-CANDIDATE ISSUE SELECTION using the L^1 norm. Observe that for a fixed pair $c_{1,j}$ and $c_{2,j}$, $\left|c_{1,j} - v_{i,j}\right|^p - \left|c_{2j} - v_{i,j}\right|^p$ is continuous for any choice of $v_{i,j} \in [c_{1,j}, c_{2,j}]$. So, $\forall$ F, we can derive a V and C that gets us F. □

Inapproximability follows directly from our reduction of MIS to TCMS: we know that MIS is NP-hard to approximate within any constant factor $c > 0$ [29], and our reduction from MIS is approximation-preserving. $\square$

The next results show that the worst-case tie breaking setting is no easier than when ties are broken in c_1's favor.

Theorem 3.3.6. *The worst-case version of* TCIS *is at least as hard as the best-case version of* TCIS *for the two-candidate case.*

Proof Sketch. Given an $n \times \ell$ matrix M associated with a two-candidate instance of best-case issue selection, define ε as in the proof of Theorem 3.3.2. Further, we let $x := \max_{j\in[1:n],k\in[1:\ell]} \left|M_{j,k}\right|$, and create a $3n \times (\ell+1)$ matrix M' as follows:

$$
\begin{aligned}
M'_{j,k} &:= M_{j,k} && j \in [1:n] && k \in [1:\ell] \\
M'_{j,k} &:= x && j \in [n+1:2n] && k \in [1:\ell] && (3.9)\\
M'_{j,k} &:= -x && j \in [2n+1:3n] && k \in [1:\ell] && (3.10)\\
M'_{j,\ell+1} &:= \frac{\varepsilon}{2} && j \in [1:n] \\
M'_{j,\ell+1} &:= -\frac{\varepsilon}{2} && j \in [n+1:3n]
\end{aligned}
$$

Once again, we choose a value of $\varepsilon > 0$ such that ε will affect the election only if a voter is undecided. The proper assignment is shown in the supplement.

Recall that in the worst-case version of TCIS, undecided voters (rows of M' with a net zero value) will default to a candidate other than c_1. With the addition of column $\ell+1$, any undecided voters will now be "nudged" in the direction of c_1 instead. Also, since the values of column $n+1$ are smaller than the difference of any two values of M, the issue affects the election only if a voter is actually undecided. So, issue $\ell+1$ appropriately mimics the weak inequality used in the best-case version of TCIS, and if a candidate wins an election in the worst-case reduction, they win the election in the best-case version, and vice versa.

Note: we add $2n$ extra voters to the problem to set things up such that including issue $n+1$ would not be sufficient for winning the election. We also choose $2n$ voters specifically so that we can be guaranteed to split voters evenly between c_1 and c_2 with our assignments in Equations 3.9 and 3.10. □

Corollary 3.3.7. *The worst-case version of* TCMS *is NP-hard.*

3.4 Binary Issues

We have shown that election control through issue selection is hard in general. However, real world opinions may have a variety of restrictions. For example, legislative issues can be viewed as *binary issues*, where a voter opinion can take only two values: support or oppose.

Formally, in binary versions of the issue selection problems, $X = \{0,1\}^\ell$. Voters vote for the candidate with whom they agree on most issues. Let BINARY ISSUE SELECTION CONTROL (BISC) be the variant of over a binary domain and, similarly, let BINARY MAX SUPPORT (BMS) be the corresponding variant of the MAX SUPPORT problem.

3.4.1 Binary Issue Selection with 1, 2 and 3 Voters

We start by considering again the problem of issue selection with a single voter, which we showed to be NP-Hard in the general case of real-valued issues. We show that this problem is now in P.

As before, it suffices to consider solely SINGLE-VOTER . We start with the case when ties are broken in c_1's favor (best-case tie-breaking). Consider the following SINGLE ISSUE WIN algorithm:

Check if there is an issue such that either (a) c_1 agrees with the voter v, or (b) no other candidate c_j agrees with v. If it exists, return YES. Otherwise, return NO.

Theorem 3.4.1. *The* SINGLE ISSUE WIN *algorithm solves* SINGLE-VOTER *with best-case tie-breaking.*

Proof. It suffices to show that whenever SINGLE ISSUE WIN returns NO, c_1 cannot win the election. Consider an arbitrary subset of issues S. Since the answer is NO, it must be that for each issue $k \in S$, c_1 disagrees with v on k. Consequently, $\|v - c_1\| = |S|$. Choose a c_j which agrees with v on some issue $k \in S$. Then $\|v - c_j\| \leq |S| - 1$, that is, c_1 cannot win for issues restricted to S. Since S is arbitrary, the result follows. □

In fact, we can easily generalize the algorithm for a single voter to a setting with two voters by simply applying the algorithm for each voter.

Corollary 3.4.2. 2-VOTER *problem with best-case tie-breaking is poly-time solvable.*

Next, we show that the problem is in P for one and two voters even with worst-case tie-breaking, although the algorithmic approach is quite different. For worst-case tie-breaking, we propose the following AGREE ON ISSUES algorithm:

Let S_{agree} be the set of all issues on which c_1 agrees with v. If c_1 wins over each other candidate c_j when issues are restricted to S_{agree}, return YES. Otherwise, return NO.

Theorem 3.4.3. *The* AGREE ON ISSUES *algorithm solves* SINGLE-VOTER *with worst-case tie-breaking.*

Proof. It suffices to consider the case when we return NO. Suppose there is some c_j that wins when we restrict to S_{agree}. Then it must be that c_j also agrees with v on all issues in S_{agree} (and any subset thereof). Consider an arbitrary subset of issues S, and let $x_{jk} = 1$ if j agrees with v on issue k. c_j's difference from v is then $\sum_{k \in S \cap S_{agree}} x_{jk} + \sum_{k \in S - S \cap S_{agree}} x_{jk} \geq |S \cap S_{agree}|$. Since the difference between c_i and v is $|S \cap S_{agree}|$, the result follows. □

The same approach is also applicable to 2-VOTER BIS.

Corollary 3.4.4. 2-VOTER *with the wost-case tie-breaking is poly-time solvable.*

Proof. For the candidate c_1 to win, **both** voters must support her. Without loss of generality, we can assume that c_1 opinion on all isues is 1. Let S_{agree} be the set of all issues on which c_1 agrees with both voter v_1 **and** v_2. Similarly to Theorem 3.4.3, if c_1 does not win against each other candidate c_j over the set S_{agree}, then no other subset of issues will achieve c_1's win. □

Remarkably, while BSIC with 1 and 2 voters are efficiently solvable for both best-case and worst-case tie-breaking, with 3 voters we see a qualitative difference in complexity, depending on how ties are broken. First, we observe that the 3-voter case with worst-case tie-breaking is tractable.

Corollary 3.4.5. 3-VOTER BINARY ISSUE SELECTION *with the worst-case tie-breaking is poly-time solvable.*

Proof. By Corollary 3.4.2 we can test in poly-time whether any given pair of voters can be won over by c_1. Applying this to each of the three possible pairs of voters, we can determine in poly-time whether the support of any two voters can be obtained simultaneously. If so, then c_1 can be made to win. Otherwise no subset of issues will make c_1 the winner. □

Now, we show that the problem becomes hard with best-case tie-breaking even with only 3 voters.

Theorem 3.4.6. 3-VOTER BINARY ISSUE SELECTION *with the best-case tie-breaking is NP-hard.*

Proof. The proof relies on a reduction from the EXACT 3-COVER (X3C) problem. An instance of X3C is governed by t – number of elements, s – the number of sets. In the reduced instance we will denote by w the preferred candidate (and assume that his opinion on all issues is 1), c – the candidate whose opinion on every issue is 0 (zero), v_3 – the voter whose opinion on every issue is 0. This implies that to win the election w should gain the support of both voters v_1 and v_2. In addition we will denote by r the number of issues in the reduced instance, setting it to $r = s+t+2$. Finally, we will set the number of candidates to $m = t+4$ and name them $c_1, \ldots, c_t, x, y, c, w$.

The preferences of v_1 and v_2 over the r issues are as follows:
$$\begin{array}{lcccc} v_1: & 1\ldots1 & 0\ldots0 & 1 & 0 \\ v_2: & \underbrace{0\ldots0}_{s} & \underbrace{1\ldots1}_{t} & 0 & 1 \end{array}$$

Preferences of candidates take a more complex form

- *For issues from* 1 *through* s. These preferences will encode the X3C instance. In particular, candidates c_i, c_j, c_e will have opinion 1 on the k^{th} issue if and only if the k^{th} set in the X3C instance is $\{i, j, e\} = S_k$. Otherwise the opinion of these three candidates on the k^{th} issue will be 0 (zero).
- *On issues* $s+1$ *through* $s+t$ all candidates $c_1, \ldots, c_t$ have 0 (zero) opinion.
- *On the* $s+t+1$ *issue* all candidates $c_1, \ldots, c_t$ have opinion 0 (zero)
- *On* $s+t+2$ *issue* all candidates $c_1, \ldots, c_t$ have opinion 1
- *Candidate* y has opinion 1 on issues $1, \ldots, s+t$ and opinion 0 (zero) on the issues $s+t+1$ and $s+t+2$
- *Candidates* x has opinions in the complete opposion to candidate y

Let us now show that if we have a solution to the resulting problem, we can recover a solution for the original X3C instance.

Candidate c, with all his opinions set to 0 (zero), serves as a kind of reference for voters. Thus, given a selection S of issues, the preferred candidate w will gain the support of a voter only if they agree on at least as many issues in S as they disagree. As a result, solution should contain equal number, q, of issues from the set $\{1, \ldots, s, s+t+1\}$ and from the set

$\{s+1,\ldots,s+t,s+t+2\}$. Consequently, candidate w will agree with any voter on exactly q issues.

Notice that both the issue $s+t+1$ and $s+t+2$ must be selected in a solution to the . To see this consider the follwoing two cases

- *Neither $s+t+1$, nor $s+t+2$ are in the solution set, S of issues.* Still, an equal number of elements (denoted earlier by q) must be selected from the sets of issues $\{1,\ldots,s\}$ and $\{s+1,\ldots,s+t\}$ for the solution set S. Wlog., issue $1 \in S$. Then voter v_1 agreed with the candidate c_{i_1} on $q+1$ issues (q issues from the set $\{s+1,\ldots,s+t\}$ and issue 1). As a result, voter v_1 would *not* vote for candidate w. Thus S, that does not contain neither $s+t+1$ nor $s+t+2$, can not be a valid solution to our instance.

- *Only one among issues $s+t+1$ and $s+t+2$ is selected as a part of the solution set of issues S.* If it is the issue $s+t+1$, then voter v_1 agreed with the candidate x on $q+1$ issues and with candidate w on q issues only. Thus, v_1 would not vote for w, and S is not a valid soluion. Similarly, if $s+t+2$ was selected, then candidate y will win the support of v_2, once again preventing w from winning.

Now, with both issues $s+t+1$ and $s+t+2$ chosen, let us show how we can obtain a solution to the original X3C problem from the solution set of issues S to the reduced problem. The set of issues S makes candidate w the winner of the election. Let $\{i_1,\ldots,i_{q-1}\} = S \cap \{1,\ldots,s\}$. We will show that the collection $S_{i_1},\ldots,S_{i_{q-1}}$ is the solution to the original X3C instance.

1. If *there is an element j that belongs to two different sets in the collection $S_{i_1},\ldots,S_{i_{q-1}}$*, then v_1 agrees with c_j on 2 issues from $i_1,\ldots,i_{q-1}$ and on $q-1$ issues from $\{s+1,\ldots,s+t\}$. Totalling $q+1$ agreements between v_1 and c_j. Which implies that v_1 will *not* vote for w, and contradicts w being the winner.

2. If *there exists an element j that does not belong to any set in the collection $S_{i_1},\ldots,S_{i_{q-1}}$*,

then $c_j \in C \setminus \{c_{j_i}, c_{k_i}, c_{e_i}\}$ for all $i \in \{i_1, \dots, i_{q-1}\}$. As a consequence v_2 agrees with c_j on $q-1$ issues from the set of issues $\{1, \dots, s\}$ and on both issues $s+t+1$ and $s+t+2$. This totals $q+1$ agreements between v_2 and c_j, entailing that v_2 will *not* vote for w, contradicting w being the winner.

As a result, the collection $S_{i_1}, \dots, S_{i_{q-1}}$ constructed from the solution S is a proper solution to the original X3C instance, i.e. every element belong to 1 and only 1 set.

Let us now show that a solution to the X3C instance can be translated into a solution to the reduction instance.

Let $S_{i_1}, \dots, S_{i_k}$ be a legal solution to the X3C instance. Then set the selection of issues $S = \{i_1, \dots, i_k\} \cup \{s+1, \dots, s+k\} \cup \{s+t+1, s+t+2\}$. Notice that k is the number of elements in the X3C instance, and therefore $k = \frac{t}{3}$ and $s+k < s+t$.

By the choice of $i_1, \dots, i_k$, it must hold that v_1 agrees with every candidate c_j once on issues $i_1, \dots, i_k$ and $\frac{t}{3}$ times on issues $s+1, \dots, s+k, s+t+1, s+t+2$. Overall v_1 and c_j agree on $\frac{t}{3}+1$ issues. Candidate x agrees with v_1 on issues $s+1, \dots, s+k, s+t+1$ only, totalling $\frac{t}{3}+1$ agreements as well. Similarly, candidates c and y rake in $\frac{t}{3}+1$ agreements. Thus, by the tie-breaking rule, v_1 votes for w.

Similarly, v_2 is matched with the opinion of c_j over $\frac{t}{3}-1$ issues from the set $\{i_1, \dots, i_k\}$ and 2 more matches are produced over issues $s+t+1, s+t+2$. This totals $\frac{t}{3}+1$ matches between c_j and v_2. Similarly to v_1, v_2 also agrees with x, y and c on $\frac{t}{3}+1$ issues. Again, tie-breaking will decide in favour of w. Thus w has the support of both v_1 and v_2 and becomes the winner.

We conclude that the original X3C instance has a solution if and only if the reduction instance of has a solution. $\square$

3.4.2 Binary Issue Selection with Two Candidates

With an arbitrary number of voters and only two candidates, even the problem with best-case tie-breaking is hard.

Theorem 3.4.7. *With two candidates, with best-case tie-breaking is NP-complete.*

Proof. It is evident that problem is in NP, so we only need to show that it is NP-hard. We will do so by a reduction from HITTING SET, where p denotes the number of elements, s – the number of sets, and k — the number of elements which should be chosen as the hitting set. We construct a profile for problem with 2 candidates, ℓ issues and n voters, where ℓ is such that $\ell = p+k$ and $n = 2ks+4$.

We assume that the preferred candidate is c_1 and set his opinion to 1 on all issues. All opinions of his rival, c_2, are set to 0 (zero). We then arrange voters into 3 blocks, as follows:

- **[Block 1.]** Two voters. The first one has opinion 0 (zero) for issues from 1 through issue $\ell-k$, and opinion 1 for issues from $\ell-k+1$ to ℓ. The second voter has an opposite opinion wrt all issues.
- **[Block 2.]** Second block consists of ks voters divided into k sub-blocks. For every sub-block, opinions of voters on issues from 1 to $\ell-k$ encode the hitting set problem instance. That is, voter $(f-1)s+i$ has opinion 1 on issue j if and only if element $j \in s_i$ for all $f \in [1:k]$. For issues from $\ell-k+1$ to ℓ, all voters of the sub-block $f \in [1:k]$ will have the same 0 (zero) opinion on issue $\ell-k+f$ and 1 on all other issues.
- **[Block 3.]** This block consists of $ks+2$ voters whose opinion on all issues is 0.

Let us now show the correctness of this reduction. Let $\{i_1,\dots,i_j\}$ be a set issues chosen to make c_1 the winner. Consider voters who support c_1. Evidently, nobody from Block 3 is

	Sub-block 1				**Sub-block 2**				**...**	**Sub-block k**			
voters/issues	1	2	$\dots$	s	$s+1$	$s+2$	$\dots$	$2s$	$\dots$	$(k-1)s+1$	$(k-1)s+2$	$\dots$	ks
1	Hitting				Hitting				$\dots$	Hitting			
2	Set				Set				$\dots$	Set			
$\vdots$	Problem				Problem				$\dots$	Problem			
$\ell-k$	Encoding				Encoding				$\dots$	Encoding			
$\ell-k+1$	0	0	$\dots$	0	1	1	$\dots$	1	$\dots$	1	1	$\dots$	1
$\ell-k+2$	1	1	$\dots$	1	0	0	$\dots$	0	$\dots$	1	1	$\dots$	1
$\vdots$	$\vdots$	$\vdots$	$\dots$	$\vdots$	$\vdots$	$\vdots$	$\dots$	$\vdots$	$\dots$	$\vdots$	$\vdots$	$\dots$	$\vdots$
ℓ	1	1	$\dots$	1	1	1	$\dots$	1	$\dots$	0	0	$\dots$	0

Table 3.1: Theorem 3.4.7. Voters' profile construction. Block 2.

among them — no matter which issues were chosen, voters from Block 3 will support c_2. As a result, c_2 has at least $ks+2$ votes. Hence, all voters from Blocks 1&2 should vote for c_1 to make him the winner.

Consider voters in Block 1. They both vote for c_1, therefore, $\{i_1, \dots, i_j\}$ consists of equal number of elements from both issue sets $[1 : \ell-k]$ and $[\ell-k+1 : \ell]$. Otherwise, there are (w.l.o.g.) more issues from $[1 : \ell-k]$ than from $[\ell-k+1 : \ell]$. Which implies that the second voter from Block-1 has more negative (0) opinions than positive (1), and he will vote for the candidate c_2. Additionally that means at most k issues were picked from both sets. Denote this number by $r \leq k$.

W.l.o.g. issue $\ell-k+1$ is chosen from the set $[\ell-k+1 : \ell]$. Thus, voters from the first sub-block of Block-2 have $r-1$ 1's and one 0 as an opinion on issues in $[\ell-k+1 : \ell]$. Therefore, all voters from this sub-block should have at least one positive (1) opinion on issues chosen from the issues set $[1 : \ell-k]$. That is, these issues represent a hitting set with r elements where $r \leq k$.

Similarly a solution for the can be constructed from a given HITTING SET solution.

This proof is easy to adapt to worst-case tie-breaking. □

Corollary 3.4.8. *The* BMS *problem is NP-hard.*

Although BINARY MAX SUPPORT is NP-hard, we now show that it is easy to achieve a $\frac{1}{2}$-approximation using the following BEST-SINGLE-ISSUE algorithm: choose one issue that maximizes the net number of voters c_1 captures.

Theorem 3.4.9. *The* BEST-SINGLE-ISSUE *algorithm approximates* 2*-candidate* BINARY MAX SUPPORT *to within a factor of* $\frac{1}{2}$*, for best-case and worst-case tie-breaking.*

Proof. Let's denote number of voters by n and the number of issues by ℓ. Among two candidates c_1 and c_2 the promoted one is c_1. Without loss of generality, we can assume that candidate c_1 has opinion 1 on every issue. We will provide proof for the case of best-case tie-breaking and will describe changes needed to transform this proof into proof for worst-case tie-breaking.

- **[Case 0.]** There is an issue s.t. candidate c_2 also has opinion 1 about this issue. Therefore, if we highlight only this issue all voters will vote for c_1 because of tie-breaking. Thus, that is an optimal solution (and as such approximation within factor 2 of optimal solution). It is also the issue that captured the greatest number of voters for c_1 if highlighted. From now on we can assume that opinion of candidate c_2 is 0 for all issues.

- **[Case 1.]** There exists an issue s.t. at at least $\frac{n}{2}$ voters have same opinion as c_1. If highlighted such issue will capture for c_1 at least $\frac{n}{2}$ voters. That is, for issue that causes c_1 to capture the greatest number of voters it is at least $\frac{n}{2}$ voters too. Thus, it is provide $\frac{1}{2}$-approximation, because optimal solution is at most n.

- **[Case 2.]** Now we can assume that for all issues less then $\frac{n}{2}$ voters have opinion 1. Denote the largest such number by h and show that h is $\frac{1}{2}$-approximation of optimum. Assume the contrary. W.l.o.g. issues $s_1, \ldots, s_k$ maximizes support for candidate c_1. By choice of h the number of opinions which equals to 1 over all issues $s_1, \ldots, s_k$ is at most kh. On the other hand voter supports candidate c_1 if and only if he has opinion 1 for at least $\frac{k}{2}$ issues among $s_1, \ldots, s_k$. By assumption there are strictly more than $2h$

such issues. That is, on issues $s_1, \ldots, s_k$ opinion 1 shared strictly more than $\frac{k}{2}2h = kh$ times. Obtained contradiction proves the theorem.

This proof can be easily adopted for the case of worst-case tie breaking. It is easy to see that if candidate c_2 has opinion 1 on all issues then for every highlighted set of issues support of candidate c_1 will be 0. Thus, any single issue provides $\frac{1}{2}$-approximation of optimum. Therefore, we may assume that there exist issue on which candidate c_2 has opinion 0.

Evidently, if there is optimum $s_{i_1}, \ldots, s_{i_k}$ such that on some of highlighted issues candidate c_2 has opinion 1. W.l.o.g. this issue s_{i_k} then $s_{i_1}, \ldots, s_{i_{k-1}}$ is also optimum. Therefore, we may assume that candidates have different opinions on all issues. Thus, it is enough to consider cases 1 and 2. The proof for case 1 remains unchanged. For case 2 we should change the counting of number of points needed to obtain at least $2h$ votes in favor of candidate c_1. A voter would only vote for c_1 if he has opinion 1 for $\lfloor \frac{k}{2} \rfloor + 1$ issues $s_1, \ldots, s_k$. Therefore, the number of opinions 1 is $(\lfloor \frac{k}{2} \rfloor + 1)2h \geq kh + h$, yielding same contradiction as in best-case tie-breaking. $\square$

3.5 Algorithmic Approaches

We now present several general algorithmic approaches for MAX SUPPORT: 1) exact approaches based on integer linear programming (ILP), and 2) a heuristic approach which works well in practice.

Integer Linear Programming: Define A as follows:

$$A_{ijk} = \left|c_{ik} - v_{jk}\right|^p - \left|c_{1k} - v_{jk}\right|^p, \forall i \in [2:m], j \in V, k \in [1:\ell]. \tag{3.11}$$

Define $\alpha := \sum_{ijk} \left|A_{ijk}\right|$. The following ILP computes an optimal solution for (best-case)

MAX SUPPORT:

$$\max_x \sum_i^m y_i \tag{3.12a}$$

$$\sum_k A_{ijk} x_k + (1 - y_j)\alpha \geq 0 \qquad \forall i \in [2:m], j \in V \tag{3.12b}$$

$$x_k, y_j \in \{0,1\} \qquad \forall k \in [1:\ell], j \in V. \tag{3.12c}$$

Constraint (3.12b), ensures that $y_j = 1$ iff c_1 is the most favored by voter j. A similar approach can be used to develop a ILP approach for the ISSUE SELECTION CONTROL problem:

$$\forall i, i' \in C, j \in V:$$

$$\sum_k A_{i'jk} x_k - \sum_k A_{ijk} x_k + (1 - z_{ij})\alpha \geq 0 \tag{3.13a}$$

$$\forall i \in C, j \in V: \sum_k A_{ijk} x_k + (1 - y_j)\alpha \geq 0 \tag{3.13b}$$

$$\forall j \in V: \sum_{i \in C} z_{ij} + y_j = 1 \tag{3.13c}$$

$$\forall i \in C: \sum_{j \in V} y_j - \sum_{j \in V} z_{ij} \geq 1 \tag{3.13d}$$

$$\forall k \in I: x_k \in \{0,1\} \tag{3.13e}$$

$$\forall j \in V: y_j \in \{0,1\} \tag{3.13f}$$

$$\forall i \in C, j \in V: z_{ij} \in \{0,1\}. \tag{3.13g}$$

Greedy Heuristic: Finally, we present a simple greedy algorithm for the MAX SUPPORT problem, where we iteratively add one issue at a time that maximizes the net gain in voters. We stop when adding any more issues would decrease the number of voters captured.

Our experimental results show that this simple greedy algorithm performs much better than BEST-SINGLE-ISSUE when it comes to real-valued MAX SUPPORT, and equivalent to Best-Single-Issue for the task of BINARY MAX SUPPORT. Furthermore, the algorithm runs pretty quickly, with a worst-case runtime of $O(n^2)$.

3.6 Experiments

We now compare the performance of our exact and heuristic solution algorithms for the binary and continuous versions of the issue selection problem. We consider the greedy heuristics described above, as well as BEST-SINGLE-ISSUE.

3.6.1 Setup

We run all of our experiments assuming a worst-case tie-breaking rule and generate random synthetic test cases. For continuous test problems, we sample candidate and voter belief vectors from the multivariate normal distribution with a mean of 0 and a random covariance matrix. A similar generative model for Boolean issues, tends to produce problem instances in which BEST-SINGLE-ISSUE is nearly always optimal.

Binary test problems are generated in a more sophisticated fashion, as uniformly sampling from the Boolean hypercube $\{0,1\}^n$ tends to produce trivial problem instances in which a single issue is almost always optimal. Consequently, we generate a more specialized distribution of these instances as follows. We first construct a vertex-weighted complete binary tree T on $2^\ell - 1$ vertices. Each vertex v is assigned an independent random weight p_v drawn from the uniform distribution on $[0,1]$. To produce a sample from T, we perform a directed random walk from its root to one of its leaves. The sequence (0 for left movements, and 1 for right) emitted by this process is then the desired sample from $\{0,1\}^\ell$.

We default to 3 candidates, 100 voters, and 10 issues. To generate each plot, we fix 2 of these parameters and vary the 3rd. We generate 100 instances of MAX SUPPORT for each set of parameter values, and run the heuristics on the instances. The plotted values are averages of the ratio of the number of voters captured and the optimal solution.

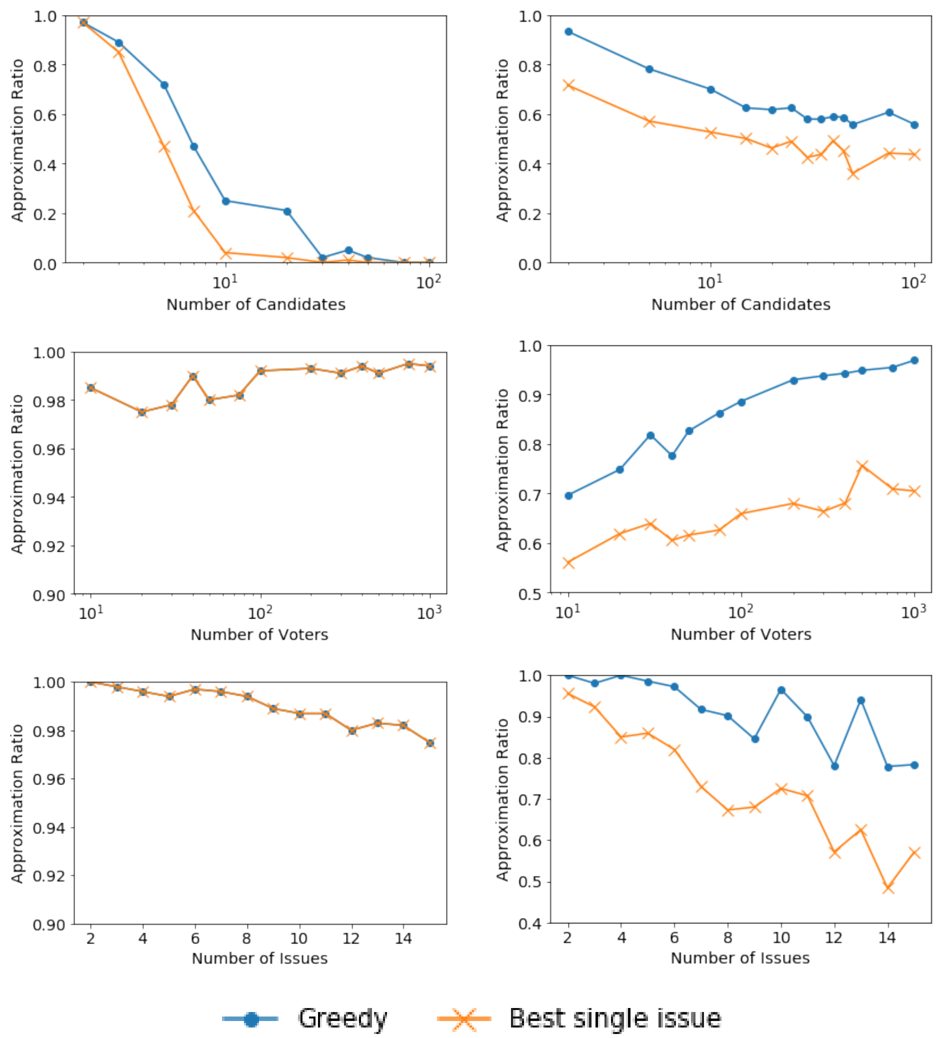

Figure 3.2: Plots of experimentally observed approximation ratios as functions of the numbers of candidates, voters, and issues in synthetic test cases for binary (left) and continuous (right) versions of MAX SUPPORT.

3.6.2 Results

We find that for most instances of MAX SUPPORT with binary issues, our greedy heuristic does not significantly outperform BEST-SINGLE-ISSUE in the two-candidate setting as number of issues and voters increase. This is because the number of instances in which a combination of issues can get us more voters than a single best issue is increasingly unlikely. However, the greedy algorithm outperforms BEST-SINGLE-ISSUE on instances of BINARY MAX SUPPORT with greater than 2 candidates. We can also observe that on the specific distribution of binary issue instances we generate, the quality of heuristic solutions degrades rapidly with the number of candidates.

We find that for MAX SUPPORT with real-valued issues, the greedy algorithm significantly outperforms BEST-SINGLE-ISSUE. For a small number of candidates (< 5), the greedy algorithm seems to perform within 0.8 of optimal. Interestingly, as the number of voters increases, the greedy algorithm improves in quality on our randomly generated problem instances. In all cases, we can also observe that the heuristics tend to be close to optimal.

Chapter 4

Point Estimation via Neural Networks

Here, we study the problem of generating point estimates of legislators using congressional roll call data. We choose to study this problem because manipulatign elections by selecting issues only becomes viable once we have produced a set of belief vectors for an electorate in the first place. Our aim is to generate vector representations of legislators in multidimensional space through the use of a technique similar to that used by neural network word embeddings. Although much research has gone into representing legislators as point estimates in one or two dimensional space, research into higher dimensional point estimation has been much more sparse. However, we believe that point estimates in higher dimensions can be much more useful in explaining legislative behavior than point estimates in just one or two dimensions.

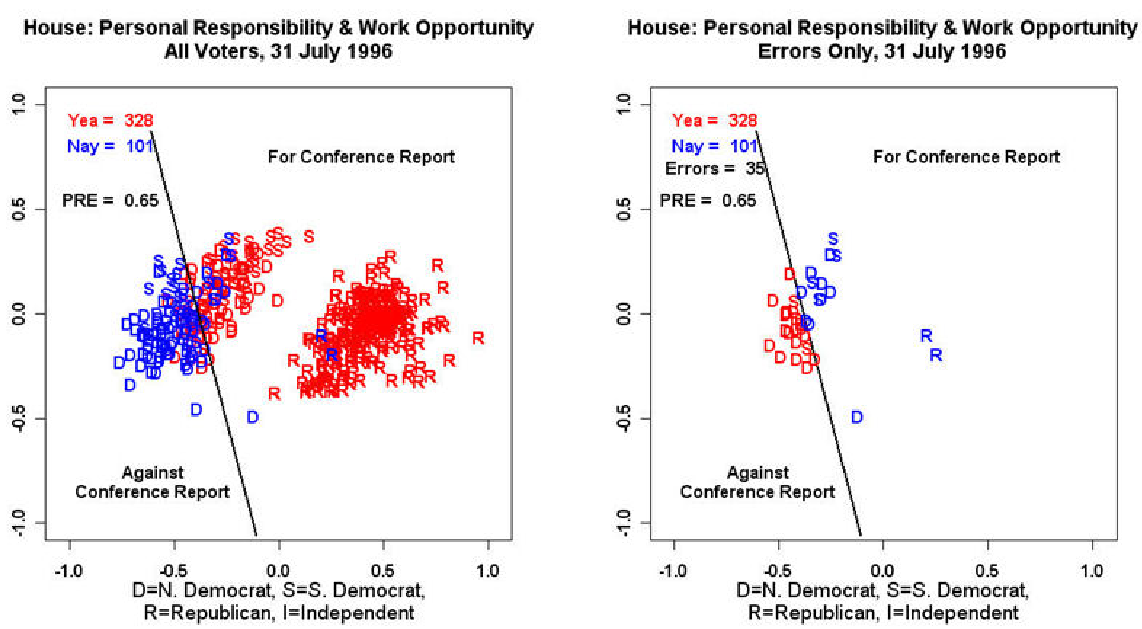

Figure 4.1: Point estimates for the U.S. House of Representatives in 1996

In this field of research, the metric used to determine how well a set of point estimates explains voting behavior is what percentage of votes over time can be recovered simply by separating the points via some hyperplane. For a better illustration of this, see figure 4.1. If we know the semantic meanings behind the X and Y axes, we can explain the house vote on the Personal Responsibility and Work Opportunity bill by saying the house made their decision based on some linear combination of the X and Y factors, $aX + bY + c$.

In the following section, we explore a technique of generating point estimates using neural networks. The primary benefit of using neural networks for this task is that We can directly investigate and compare the quality and usefulness of point estimates in different numbers of dimensions.

4.1 The Model

Assume that we are given data with n different people voting on m different issues. We can let Y be our set of issues, and let Y_{ij} index person i's vote on issue j, where $Y_{ij} = 1$ if person i votes "Yea" on issue j, and 0 otherwise. Let X be our set of voters, and represent every voters as a one-hot vector X_i. Essentially, this is a n-length vector of all 0's with a 1 at index i.

In our study here, we will be focusing primarily on the use of congressional roll-call data to generate point estimates for house representatives. Roll-call data comes as a list of voting outcomes for each house representative for a number of different bills over a two-year period of time. Even though our study is specifically on this format of data, our model can be applied to any dataset with a static list of people and a voting dataset with binary outcomes (people only have two choices for things to vote for). We discuss a method of handling outcomes such as abstentions or missing data in our experimental section later on.

Our technique uses a simple feedforward neural network model with one hidden layer.

The network will take an input vector of length n, have an arbitrary number k of nodes in its hidden layer, and output a vector of length m. The network will be trained to predict, given a particular legislator, what the legislator has voted on for each bill. To serve this purpose, we will be using a sigmoid activation function for neurons in the output layer of our network. Any arbitrary activation function can be used for the hidden layer, but we choose to use a leaky relu activation function. We choose this activation function to prevent excessive clumping of points around saturation coordinates (i.e. 0, 1 for the sigmoid activation function).

When the network is done training, we can derive point estimates of each legislator by shaving off the final layer of our network. Given an X_i, the output given by our hidden layer will be the point estimate for legislator i. Since we are using an arbitrary number of nodes, k. in our hidden layer, we can adjust the number of dimensions we want in our point estimates just by changing k.

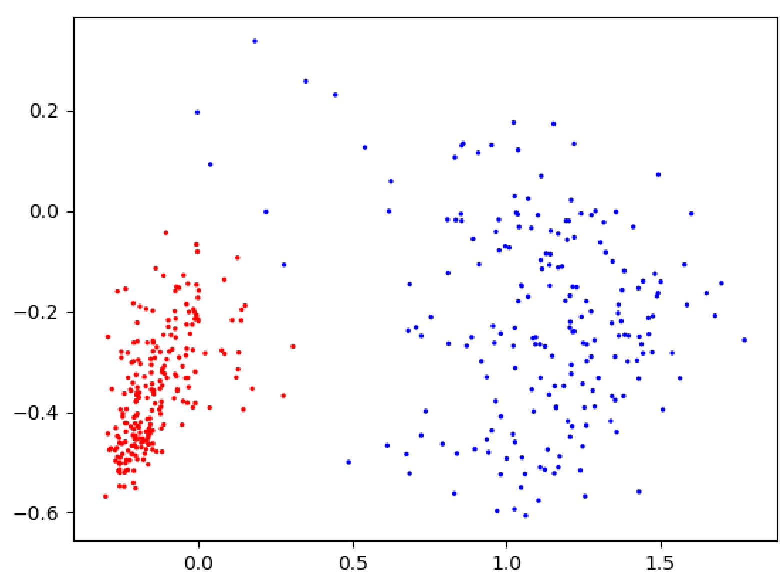

Figure 4.2: Generated point estimates for 104th congress, colored according to yea (red) or nay (blue) votes on a roll-call

Figure 4.2 gives an example of 2-d point estimates generated from congressional roll call data from the 104th congress, along with associated votes for an arbitrary call. As can be seen from the figure, we can explain the voting behavior on this roll call can be explained by simply putting a line through the point where red and blue are naturally separated. Although this may lead to just a few points being on the wrong side of the line, we have found that we are able to explain more than 90% of voting behavior for most roll calls in this way. Because a sigmoid neuron in a neural network essentially just finds an optimal separating hyperplane for a classification problem, we also end up deriving these separating lines automatically over the course of training our network.

4.2 Experimental Setup

For our experiments, we use roll call data on the 104th congress, publicly available on voteview [30]. Unfortunately, complete roll call data is unavailable due to abstentions and absent voters. Although alternative approaches to point estimation in the past have chosen to ignore such missing data, we cannot just ignore the missing data because our model assumes we possess a complete output vector with which to train with.

So, to treat missing data, we instead make two copies of the votes for each roll call. In the first copy, we set all abstentions and absent data to 0, and in the second copy we set them to 1. Our hope here is the the missing data field cancel each other out during the learning process. An alternative approach, which we do not experiment with (due to time constraints over the course of this thesis's completion) is to simply turn off backpropagation for neurons representing those roll calls we do not have voting data for.

We use pytorch 0.4.1 for our experiments. Each example is trained using an ADAM optimizer with a learning rate of 0.025 and default parameters otherwise, over 200 epochs. We also use a binary cross entropy loss function, and a leaky relu activation function with a negative slope of 0.05 for our hidden layer neurons.

4.3 Results

Our results on point estimation are mostly positive. Given a dataset of roll call data for a particular House of Congress, we are generally able to explain about 90% of congressional behavior – a slight improvement over the model developed by Poole and Rosenthal [12]. Specifically, when comparing one-dimensional point estimates on the 106th House of Congress derived using our method with that developed by Clinton, Jackman, and Rivers [13], we find that our technique can correctly classify 90.7% of individual voting decision, compared to Clinton's 89.9%.

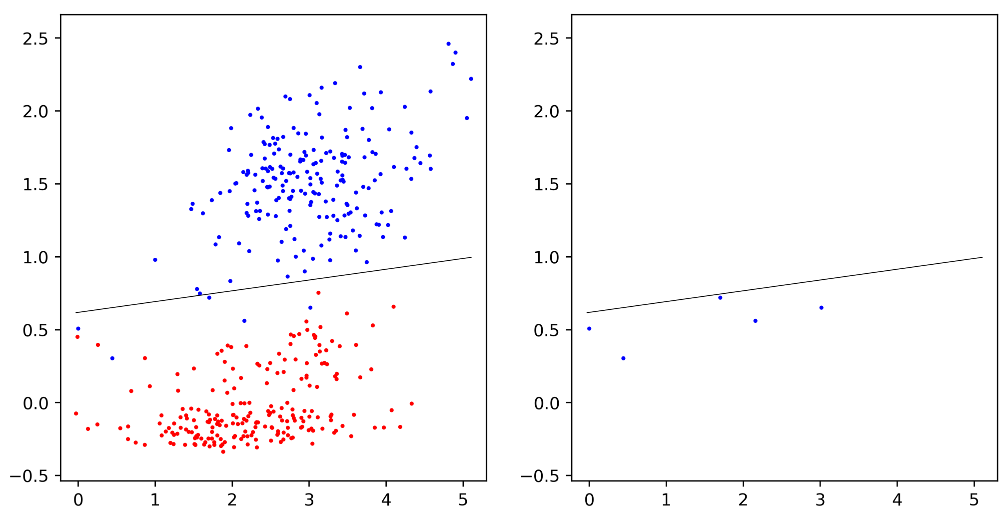

Figure 4.3: A set of point estimates on the 106th House of Congress

Figure 4.2 shows our results with two-dimensional point estimates on the 104th House of Congress, whereas figure 4.3 shows our results on the 106th House of Congress, along with the dividing line which best splits a particular roll-call vote. The left of figure 4.3 shows all of the point estimates, whereas the right image shows only those point estimates which we misclassify using our dividing line.

In initial experiments with our point estimation technique, we ended up with excessive "clumping" around the negative range of the point estimations, by nature of how the leaky

ReLu activation function works. Because of this clumping, although we can surely use these point estimates to produce classifications or predictions of voters, it is a less accurate representation of true ideologies, as this representation will end up assuming one party to be far less diverse in ideology than the other.

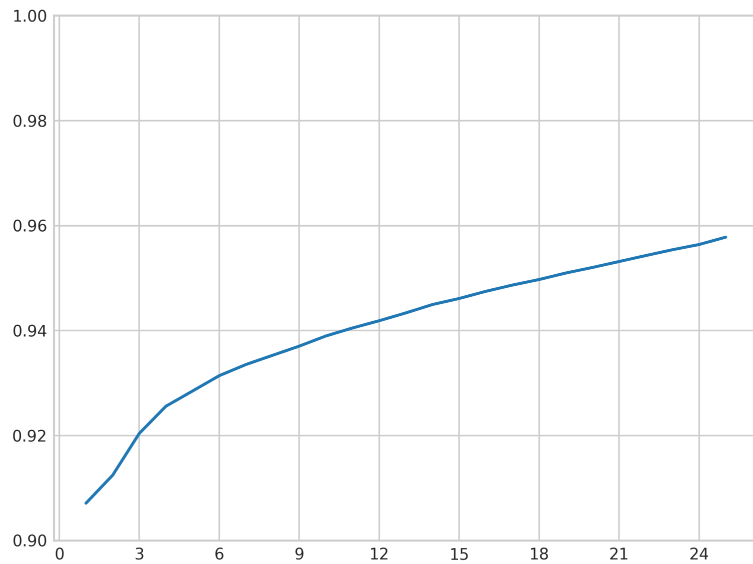

Figure 4.4: Performance of our model as we increase the dimensionality of our point estimates

In addition to performing experiments on how our point estimate model compares to those of other established techniques, we also experimented with how the dimensionality of our point estimates affected our model performance.

Figure 4.4 shows how well our model can correctly classify individual voting decisions as we increase the number of neurons in our hidden layer. Since increasing the number of neurons corresponds with increasing the dimensionality of our point estimates, we essentially use this technique to see how useful higher dimensional point estimates are in representing voters in congress.

We find that, although we get good gains in performance as we increase from 2 to 4 dimensions, our gains do not increase as drastically from 5 onward. This is in line with our hypothesis that higher dimensional point estimates can provide a better model for explaining congressional behavior than lower dimensional point estimates. However, the downside of using high-dimensional point estimates is that such models become more difficult to interpret. For example, in lower dimensions, we are better able to label our axes with educated guesses on what they might represent. In the case of the NOMINATE model, researchers often say that one axis represents economic ideology, and that the other represents social and cultural ideology. In higher dimensions, though we can represent our voters with more granular ideologies, the task of assigning meaning to the dimensions is more difficult.

Chapter 5

Conclusion

When candidates participate in an election, they must choose policies and issues to stress in their campaigns. Depending on the ideological preferences of the electorate, the policies and issues they choose can vary from election to election. This thesis has studied the problem of election control through issue election, as well as the problem of deriving point estimates of an electorate when given past voting information.

In the issue selection part of this thesis, we find a number of strong negative results for the problem, and show that, even though we cannot provide formal approximation guarantees for a continuous instance of MAX SUPPORT, a simple greedy heuristic performs well. Moreover, restricting issues to be binary admits further positive results, including a $1/2$-approximation.

In the point estimation part of this thesis, we introduce a novel technique of deriving point estimates for voters using a simple neural network. We tackle this problem because we see it as a prerequisite to constructing an instance of the Issue Selection problem: in order for a candidate to decide what issues will influence voters most in an election, they must first have knowledge of the electorate's ideological preferences. We find that, although higher dimensional point estimates can provide a better model for explaining voter behavior in elections, it remains an open problem of how to construct high dimensional point estimates in a way that we can easily assign semantic meaning to the dimensions.

Bibliography

[1] Jeff. The spatial theory of voting, 2012.

[2] Zach Hrynowski. *Several Issues Tie as Most Important in 2020 Election*, 2019 (accessed May 15, 2020).

[3] Maurice C. Bryson and William R. McDill. The political spectrum: A bi-dimensional approach. *Rampart Journal of Individualistic Thought*, 4(2):19 – 26, 1968.

[4] Harold Hotelling. Stability in competition. *The Economic Journal*, 39(153):41–57, 1929.

[5] E. Anshelevich, O. Bhardwaj, and J. Postl. Approximating optimal social choice under metric preferences. In *AAAI Conference on Artificial Intelligence*, pages 777–783, 2015.

[6] E. Anshelevich and J. Postl. Randomized social choice functions under metric preferences. In *International Joint Conference on Artificial Intelligence*, pages 46–59, 2016.

[7] Piotr Skowron and Edith Elkind. Social choice under metric preferences: Scoring rules and STV. In *AAAI Conference on Artificial Intelligence*, pages 706–712, 2017.

[8] A. Smithies. Optimum location in spatial competition. *Journal of Political Economy*, 49(3):423–439, 1941.

[9] Duncan Black. On the rationale of group decision-making. *Journal of Political Economy*, 56(1):23–34, 1948.

[10] Itay Sabato, Svetlana Obraztsova, Zinovi Rabinovich, and Jeffrey S. Rosenschein. Real candidacy games: a new model for strategic candidacy. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 867–875, 2017.

[11] Weiran Shen and Zihe Wang. Hotelling-downs model with limited attraction. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 660–668, 2017.

[12] Keith T. Poole and Howard Rosenthal. A spatial model for legislative roll call analysis. *American Journal of Political Science*, 29(2):357–384, 1985.

[13] JOSHUA CLINTON, SIMON JACKMAN, and DOUGLAS RIVERS. The statistical analysis of roll call data. *American Political Science Review*, 98(2):355370, 2004.

[14] Kyungwoo Song, Wonsung Lee, and Il-Chul Moon. Neural ideal point estimation network, 2018.

[15] John J. Bartholdi, Craig A. Tovey, and Michael A. Trick. How hard is it to control an election? *Mathematical and Computer Modelling*, 16(8):27 – 40, 1992.

[16] Edith Hemaspaandra, Lane A. Hemaspaandra, and Jörg Rothe. Anyone but him: The complexity of precluding an alternative. *Artificial Intelligence*, 171(5-6):255–285, 2007.

[17] Curtis Menton. Normalized range voting broadly resists control. *Theory of Computing Systems*, 53(4):507–531, Nov 2013.

[18] Gábor Erdélyi, Markus Nowak, and Jörg Rothe. Sincere-strategy preference-based approval voting fully resists constructive control and broadly resists destructive control. *Mathematical Logic Quarterly*, 55(4):425–443, 2009.

[19] Piotr Faliszewski, Edith Hemaspaandra, Lane A Hemaspaandra, and Jörg Rothe. Llull and copeland voting computationally resist bribery and constructive control. *Journal of Artificial Intelligence Research*, 35:275–341, 2009.

[20] Gábor Erdélyi and Jörg Rothe. Control complexity in fallback voting. In *Proceedings of the Sixteenth Symposium on Computing: the Australasian Theory-Volume 109*, pages 39–48. Australian Computer Society, Inc., 2010.

[21] Edith Hemaspaandra, Lane A Hemaspaandra, and Jörg Rothe. Hybrid elections broaden complexity-theoretic resistance to control. *Mathematical Logic Quarterly*, 55(4):397–424, 2009.

[22] Gábor Erdélyi and M Fellows. Parameterized control complexity in bucklin voting and in fallback voting. In *Proceedings of the 3rd International Workshop on Computational Social Choice*, pages 163–174. Universität Düsseldorf, 2010.

[23] Piotr Faliszewski, Edith Hemaspaandra, and Lane A. Hemaspaandra. How hard is bribery in elections? *Journal of Artificial Intelligence Research*, 35(1):485–532, 2009.

[24] Yongjie Yang, Yash Shrestha, and Jiong Guo. How hard is bribery with distance restrictions? In *European Conference on Artificial Intelligence*, 2016.

[25] Tomasz Put and Piotr Faliszewski. The complexity of voter control and shift bribery under parliament choosing rules. In *Transactions on Computational Collective Intelligence XXIII*, pages 29–50, 2016.

[26] Ujjwal Karn. A quick introduction to neural networks, 2016.

[27] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[28] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA, 2013. Curran Associates Inc.

[29] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[30] Keith Poole. 104th house page, 2017.