Missed One! How Ballot Layout and Visual Task Strategy Can Interact to Produce Voting Errors

Joshua Engels[1], Xianni Wang[2], Michael D. Byrne[1,2]
{jae4, xw48, byrne}@rice.edu
[1]Department of Computer Science, [2]Department of Psychological Sciences
6100 Main St., MS-25, Houston, TX 77005 USA

Abstract

This paper presents an ACT-R model designed to simulate voting behavior on full-face paper ballots. The model implements a non-standard voting strategy: the strategy votes first from left to right on a ballot and then from top to bottom. We ran this model on 6600 randomly-generated ballots governed by three different variables that affected the visual layout of the ballot. The findings suggest that our model's error behavior is emergent and sensitive to ballot structure. These results represent an important step towards our goal of creating a software tool capable of identifying bad ballot design.

Keywords: ACT-R; error prediction; voting

Introduction

Voting is hard. The deliberations and conversations that go into choosing who best represents one's interests is an important and time-consuming task, one that might be argued to be the very backbone of a democracy. Understandably, many may believe that the subsequent task of correctly indicating one's chosen candidate is comparatively easy and straightforward. Surely once a voter gets the ballot and can mark whoever they please, the hard part is over.

Often, this is correct. When ballots are designed well, errors voters make are not systematic and generally will not help or hurt any particular candidate. However, when ballots are designed poorly, they may lead to systematic voting errors. It is possible such errors do not matter if margins of victory are large and thus such issues may go unnoticed.

However, in closely-contested elections it is not the general case that is important. There have been numerous elections in the past 20 years that have been documented as having been decided by systematic voting errors caused by bad ballot design. This ranges from the infamous "butterfly ballot" in Palm Beach County, Florida in the year 2000 (Wand, et al., 2001) to the most recent major U.S. election in 2018, where a U.S. Senate seat (also in Florida) was almost certainly decided by a poorly-designed ballot (Chisnell & Quesenbery, 2018). For a review, see Norden, et al. (2008).

While election interference by hacking is a far more flashy and obvious risk, there has never been clear evidence that this has swung an election, unlike with bad ballot design. Ironically, the fear of hacking has led to a return to paper ballots, which with their profusion of races packed onto small sheets of paper makes ballot design even more important.

The most likely errors caused by poor ballot design are under- and overvoting. An undervote is an error that occurs when the voter fails to vote on a race that they intended to, whereas an overvote is when a voter votes on a race more than the allowable number of times (usually, more than once). The problem of designing a ballot that will not cause people to systematically under- or overvote is challenging. For instance, it might entail running a usability study weeks before the actual election. What makes the problem so difficult is the sheer number of counties in the United States (over 3000), each of which designs their ballots differently and each of which have hundreds of different iterations of ballots for each precinct they are responsible for. Manually checking each ballot with a usability study is infeasible.

One possible solution to this problem is software that could automatically check an arbitrary ballot for common design errors. However, such a solution would only find errors that had been previously made by voters on other ballots. If the task is to predict if humans will make a mistake on a novel ballot, it is difficult to imagine that chasing only known errors will be sufficient. Here is where ACT-R (Anderson, 2007) modeling comes in. Since ACT-R is generative, it can predict behavior on any ballot and is not limited only to errors that have been made previously.

Building such a predictive model is itself an extremely challenging task because it would have to be able to predict all historical voting errors as well as any new ones. For example, while Green (2010) built an ACT-R model that could make the same mistake voters did in a specific famous ballot (the 2006 Sarasota County ballot), it was limited to replicating one specific error behavior.

Thus, Wang, Lindstedt, and Byrne (2019) present the outline for an ambitious project: a model that can simulate the entire space of possible voting behaviors. They presented a smaller scale version of this end goal model. The model ran in a voting environment called VoteBox, a simulated electronic ballot, consisting of a single race per screen and a "next" button to navigate.

Nevertheless, within just this simple task was hidden great complexity: the model used a total of 40 different voting strategies constructed from differing levels of ballot/candidate knowledge and navigational strategy selections. The voters differing strategies and knowledge led to different rates of error, showing that a model voter's strategy made a difference on whether or not it was able to accurately vote for its intended candidates. However, this effort was preliminary in that it did not vary the design of the ballot; it simply demonstrated that errors were emergent from a particular combination of task strategy and memory contents.

In this paper, we describe a model that represents the natural extension of this system to show that errors can emerge from the interaction of strategy and ballot design. This model also works in a more challenging visual environment: it handles simulated full-face paper ballots. A full-face ballot is one that has all the races on a single display (usually a piece of paper). This extension introduces new model building challenges. Our new models must navigate both between and within races, and our model creation process must be flexible enough to explore an even larger voting strategy space.

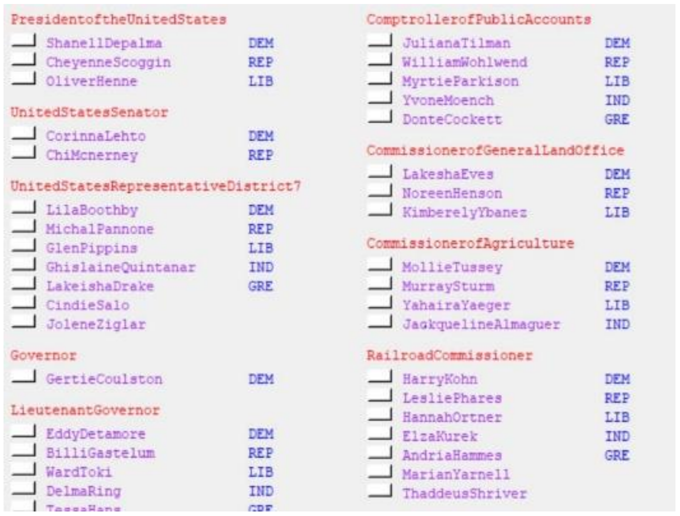
Unsurprisingly, the increased complexity of a full-face paper ballot leads to increased model error. Thus, we also describe the error rates of simulated voters on differing simulated ballots. This represents an important step towards our end goal of constructing a generative model able to identify bad ballots.

Method

First, we will describe the design of our full-face ballots, then the design of the model, and our simulation of the model across many possible ballot designs.

Ballot Design

We built simulated full-face paper ballots for the model which consist of a virtual screen populated with several columns of races. Each race has a title, a list of candidates and their associated parties, and a list of buttons that the model can click to vote for a candidate. (see Figure 1).



```
PresidentoftheUnitedStates                    ComptrollerofPublicAccounts
  ShanellDepalma          DEM                   JulianaTilman           DEM
  CheyenneScoggin         REP                   WilliamWohlwend         REP
  OliverHenne             LIB                   MyrtieParkison          LIB
                                                YvoneMoench             IND
UnitedStatesSenator                             DonteCockett            GRE
  CorinnaLehto            DEM
  ChiMcnerney             REP                 CommissionerofGeneralLandOffice
                                                LakeshaEves             DEM
UnitedStatesRepresentativeDistrict7             NoreenHenson            REP
  LilaBoothby             DEM                   KimberelyYbanez         LIB
  MichalPannone           REP
  GlenPippins             LIB                 CommissionerofAgriculture
  GhislaineQuintanar      IND                   MollieTussey            DEM
  LakeishaDrake           GRE                   MurraySturm             REP
  CindieSalo                                    YahairaYaeger           LIB
  JoleneZiglar                                  JackquelineAlmaguer     IND

Governor                                      RailroadCommissioner
  GertieCoulston          DEM                   HarryKohn               DEM
                                                LesliePhares            REP
LieutenantGovernor                              HannahOrtner            LIB
  EddyDetamore            DEM                   ElzaKurek               IND
  BilliGastelum           REP                   AndriaHammes            GRE
  WardToki                LIB                   MarianYarnell
  DelmaRing               IND                   ThaddeusShriver
```

Figure 1: Top left corner of a simulated ballot.

The resulting simulation is not quite the same as an actual paper ballot. For example, the model clicks on a button instead of filling in a circle and does not obscure the ballot with its hand while doing so. However, the ballot is typical in visual layout, which we believe is similar enough to cause many of the same errors we expect human voters to make.

Because ACT-R's nascent ability to group visual items is somewhat limited (Lindstedt & Byrne, 2018), we had to work around this. So, to help the model navigate, we colored the race header red, the candidates purple, and the parties blue. The coloring allows the model to make visual location requests like "the closest red text in the column to the right" (when finding the closest race) or "the closest purple text to my current position" (when finding the candidate group of the currently attended race). Since we suspect humans can also reliably differentiate between race headers, candidates, and parties by using the visual characteristics of the ballot, we believe coloring the ballot does not give the model an unfair advantage. However, we are exploring alternative ways to work around this problem.

Model Design

We built the model with one overarching goal in mind: to simulate as wide an array of voters as possible.

Our modular system split a simulated voter's strategy into four different pieces: (1) macronavigation, the process of moving from one race to the next; (2) visual encoding, the process of determining the race, party, and candidate visual groups for each race; (3) micronavigation, the process of finding the intended candidate to vote for within each race; and (4) selection, the process of actually clicking on the button corresponding to the chosen candidate. At runtime we selected one strategy from each of these categories and combined them together with a declarative memory file to build an ACT-R model. Note that how the model does pieces 2–4 was taken directly from the Wang et al. (2019) model.

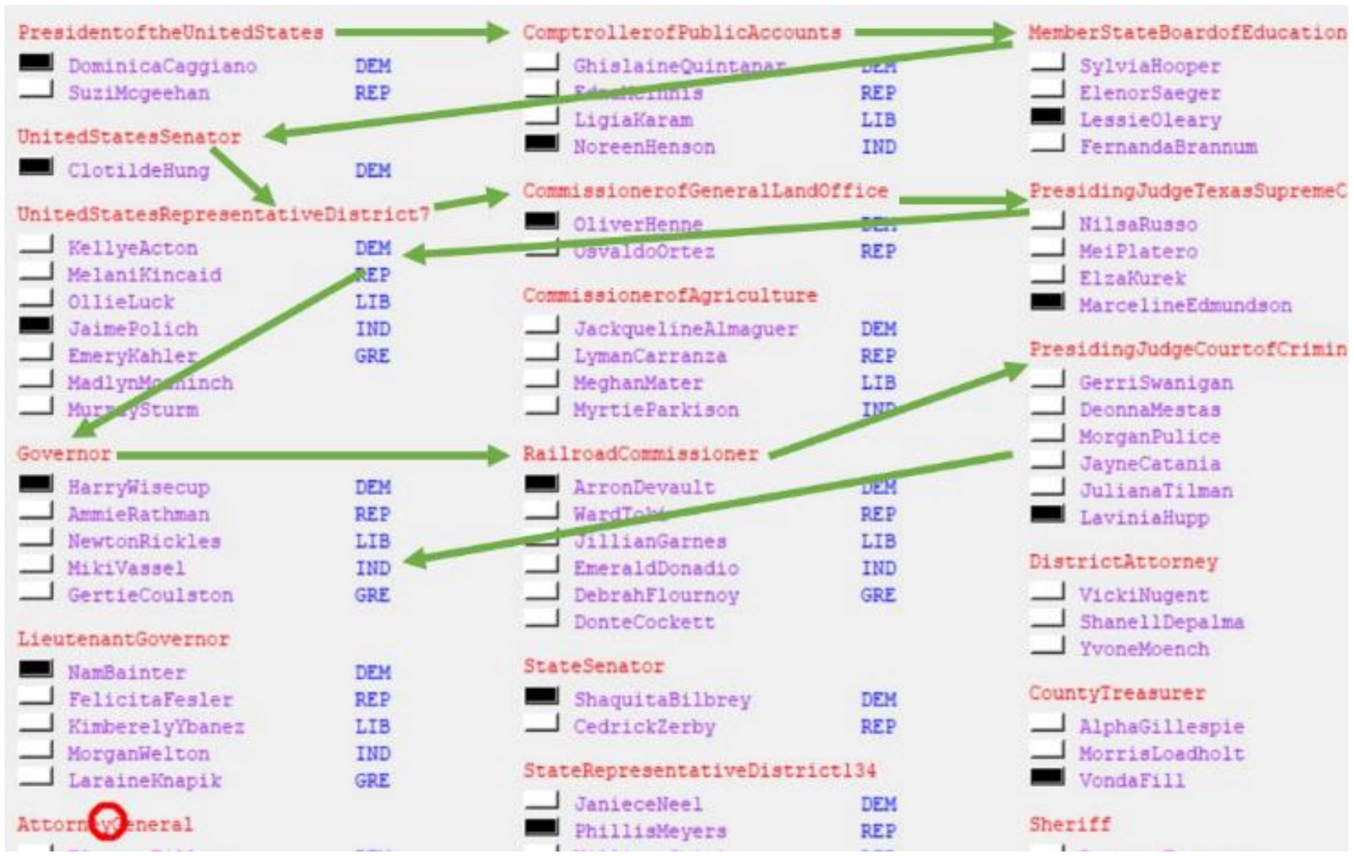
Designing A New Strategy

We first built the most obvious option for each strategy category because we wanted our initial strategies to lead to a composite voting strategy with no errors. We wanted to ensure that our model worked before we started varying pieces to induce errors.

Our first strategy after these obvious ones was a non-standard macronavigation strategy. Our model's standard macronavigation strategy was *top to bottom left to right*; that is, the model started in the top left corner and went all the way to the bottom of the column and then went over to the next column to the right and again went top-to-bottom, repeating until it was finished. This is the most obvious method of macronavigation, and as noted above resulted in no mistaken votes. The first alternative macronavigation strategy we built was *left to right top to bottom*.

The *left to right top to bottom* strategy starts on the upper leftmost race on the ballot. It then proceeds to the right, navigating to the closest race to the last race it voted on in the next column over, and repeating until it votes on a race in the last column. Then, it goes back to the beginning of the row, finds the next race down in the left column, and repeats voting from left to right. The model continues until it runs out of new races in the left column.

On our ballots the races in each column are horizontally aligned, as might be expected. However, when race lengths


PresidentoftheUnitedStates
■ DominicaCaggiano DEM
□ SuziMcgeehan REP
UnitedStatesSenator
■ ClotildeHung DEM
UnitedStatesRepresentativeDistrict7
□ KellyeActon DEM
□ MelaniKincaid REP
□ OllieLuck LIB
■ JaimePolich IND
□ EmeryKahler GRE
□ MadlynM[illegible]inch
□ Mur[illegible]Sturm
Governor
■ HarryWisecup DEM
□ AmmieRathman REP
□ NewtonRickles LIB
□ MikiVassel IND
□ GertieCoulston GRE
LieutenantGovernor
■ NamBainter DEM
□ FelicitaFesler REP
□ KimberelyYbanez LIB
□ MorganWelton IND
□ LaraineKnapik GRE
AttorneyGeneral

ComptrollerofPublicAccounts
□ GhislaineQuintanar DEM
□ [illegible]cinnis REP
□ LigiaKaram LIB
■ NoreenHenson IND
CommissionerofGeneralLandOffice
■ OliverHenne DEM
□ OsvaldoOrtez REP
CommissionerofAgriculture
□ JacquelineAlmaguer DEM
□ LymanCarranza REP
□ MeghanMater LIB
□ MyrtieParkison IND
RailroadCommissioner
■ ArronDevault DEM
□ WardTo[illegible] REP
□ JillianGarnes LIB
□ EmeraldDonadio IND
□ DebrahFlournoy GRE
□ DonteCockett
StateSenator
■ ShaquitaBilbrey DEM
□ CedrickZerby REP
StateRepresentativeDistrict134
□ JanieceNeel DEM
■ PhillisMeyers REP

MemberStateBoardofEducation
□ SylviaHooper
□ ElenorSaeger
■ LessieOleary
□ FernandaBrannum
PresidingJudgeTexasSupremeC
□ NilsaRusso
□ MeiPlatero
□ ElzaKurek
■ MarcelineEdmundson
PresidingJudgeCourtofCrimin
□ GerriSwanigan
□ DeonnaMestas
□ MorganPulice
□ JayneCatania
□ JulianaTilman
■ LaviniaHupp
DistrictAttorney
□ VickiNugent
□ ShanellDepalma
□ YvoneMoench
CountyTreasurer
□ AlphaGillespie
□ MorrisLoadholt
■ VondaFill
Sheriff

Figure 2: The green arrows mark the first part of the *left to right top to bottom* model's voting pattern on this specific ballot. The model skips CommisionerofAgriculture.

are allowed to vary, races in different columns are not *vertically* aligned, as the generation process always placed each race a set distance below the last race. Because our new macronavigation strategy proceeded initially from left to right, when races were vertically misaligned our model *could miss races*. Note that when the ballot is a perfect grid where all races are vertically aligned, the model does *not* make errors. It is the interaction of this strategy with the design of the ballot that results in errors. For an example of the model missing a race on a typical ballot, see Figure 2.

In Figure 2, when the model reaches the third race down in the left column ("United States Representative District 7") it votes on that race and then proceeds right along the row, selecting and voting on the closest race and repeating until it reaches the last column. The model then returns to the race at the beginning of the row and proceeds to the first race on the next row down ("Governor"). Here is where it makes its mistake: because the "Railroad Commissioner" race is the closest race to "Governor," the model votes on "Railroad Commissioner" for its second race in the row and so skips Commissioner of Agriculture. It never returns and votes on this race.

We observed that the races our new strategy missed depended on the layout of the races on the ballot and determined it was critical to understand if this was systematic.

Experiments

Once we had a simulated voter making structure-based mistakes, we decided to test how these mistakes changed as a function of the ballot layout. Initially, our ballot was static, consisting of a manually-positioned set of races and candidates. Our first step was modifying the ballot so it could be dynamically generated. Every time we ran the model, our generation process allowed us to vary the vertical spacing between races, the vertical space between the race header and the candidates, and the vertical space between candidates. We chose ranges of the variables that led to ballots our model could still realistically parse but that nevertheless were visually distinct (see Table 1). As the ballot was generated each race was randomly selected to have between 1 and 4 candidates.

Table 1: Ballot Layout Variables

Variable	Range (Pixels)
Space between races	5 – 15
Space between header and candidates	20 - 22
Space between candidates	15 - 18

For each one of the 132 possible combinations of spacing variables (see Table 1), we ran the model on 50 randomly generated ballots. Thus, our model was run on 6,600 ballots for a total count of 158,338 individual races. For each run, we recorded the exact race positions and race order on the ballot, as well as the order the model voted on races (including any races the model missed).

The data allow us to characterize this strategy and identify how and where it fails. We will also describe good and bad ballot design by seeing which designs lead to more error in the model. This will serve as a case study for how new strategies built on our architecture will find errors in novel ballots.

Results

First, we define model *percent error*, the percent of races that our model skips. Our model's *global* percent error is around 13.04%, meaning that, on average, given a random race on a ballot there is a 13.04% chance that our model will not vote on it. This rate is certainly much higher than any experimental rate in human voters, but as this strategy is nonstandard, this result is to be expected. Of course, most people do not make anywhere near these many errors, but average error rates in the wild likely stem from outliers like this strategy.

Effects of Race Location

We first examine the relationship of race location on the ballot to model error. We observe that there is a general trend of increasing error across columns (see Figure 3). In other words, races in columns that are further to the right are more likely to be skipped.

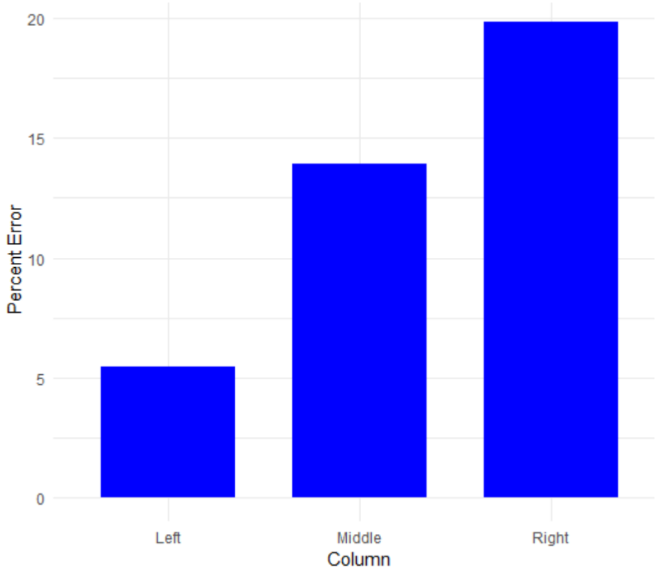


Figure 3: Average percent error across races in the left, middle, and right column across all ballot runs

In fact, since we recorded the exact *y* coordinate and column for every race on every ballot, we can generate a heatmap of error rate by race position on the ballot (see Figure 4). Each bin collates the percent error of the model on races within 10 vertical pixels, where the *y* position of a race is its header.

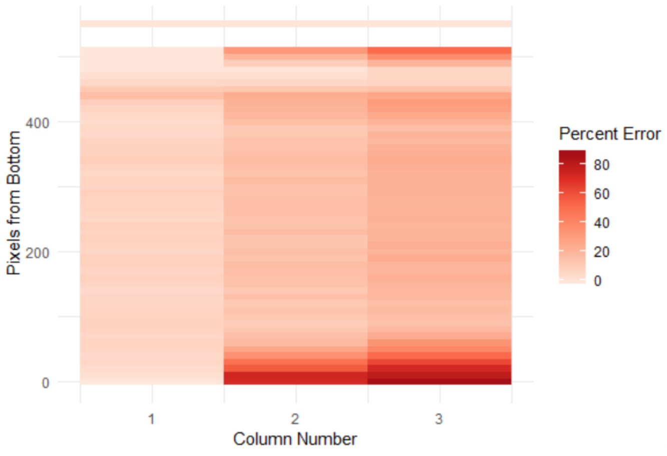


Figure 4: Heatmap of the model's error according to races' column number and *y* position.

Of interest are places in Figure 4 where errors are likely. One immediately obvious place is the bottom right corner, where average percent error approaches 1. The model almost always misses races there. To make sense of this result, we observe that the only way a race can have its *start* in one of those bottom right boxes is if it is very short. It makes sense that for short races nestled in the bottom corner, people will frequently get to the last race in the left column and vote across that row not *low enough* to reach the bottom corner races.

However, other than this, errors are more or less uniformly distributed across the ballot. This result hints at the strength of our model: errors occur seemingly randomly across the ballot because they are emerging from the specific structure of individual random ballots. Thus, using our data of each experiment's race layout, we move onto examining how specific elements of ballot structure influence model error.

Effects of Ballot Structure

We first examine the error rate as we vary the amount of vertical space between the end of each race and the beginning of the next. Recall that vertical space is just one of the spacing variables we manipulated (see Table 1). Thus, each specific vertical spacing value includes many observations from ballots built from combinations of the other spacing variables. While we did examine these other spacing variables, we found they had no significant effect on the model's error rate.

As the space between races decreased, voting error increased (see Figure 5). This result validates the intuition that the more cluttered a ballot is, the more likely a simulated voter is to miss a race.

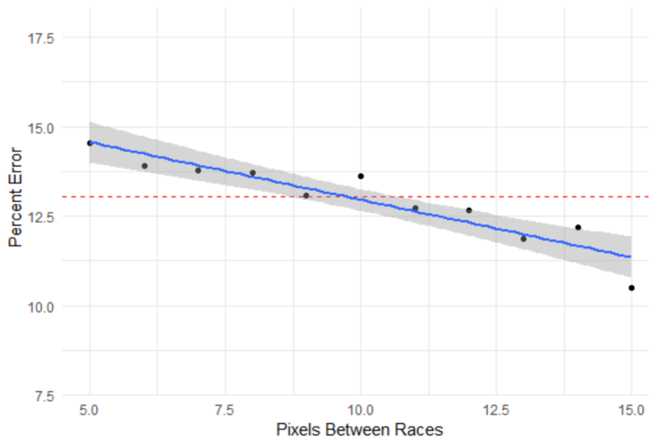


Figure 5: Each black dot is the average percent error across all ballots with a specific race spacing. The blue line is the linear regression for the trend, the red line is the average error of the model, and the shading represents 95% confidence intervals for the line.

We also examined how the length of a race was related to the chance it would be skipped and found similar results: as the length of a race decreased, the model's chance of skipping it (its error rate for races of that length) increased (see Figure 6). Of note, single-candidate races are most likely to be missed, but of course skipping such a race will not change the outcome of an election, since unopposed candidates are guaranteed to win.

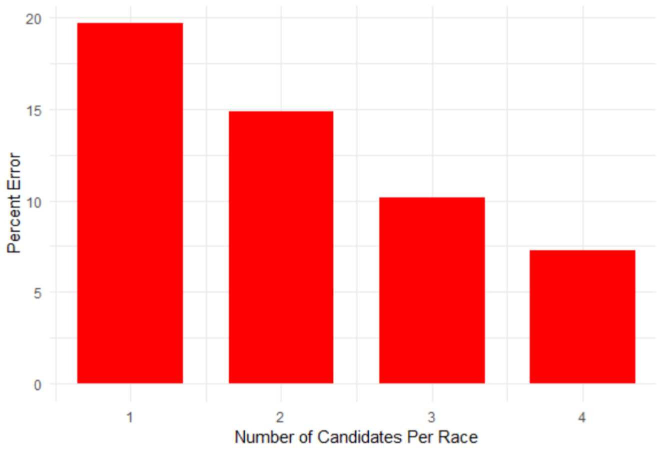


Figure 6: Average error rate of the model on races of one candidate, two candidates, three candidates, and four candidates.

Finally, we looked at how the model's error rate varied as a function of the vertical distance from a given race to the nearest race to it in the last column. In Figure 7, we show a stacked bar plot of races missed and races voting on according to this variable. This graph shows two things: one, that the chance a simulated voter misses a race increases as the closest distance to the last race increases, and two, that the number of races that are far from any prior race decreases as the distance increases. The reason that the distribution is non uniform, with peaks in the 0 bin, 15-20 bin, 30-35 bin, and 45-50 bin, is a result of the way ballots were generated. The candidate spacing varied from 15 to 18 pixels (see Figure 2), and it was frequently the case that the closest race in the last column was an integer multiple of candidate space away.

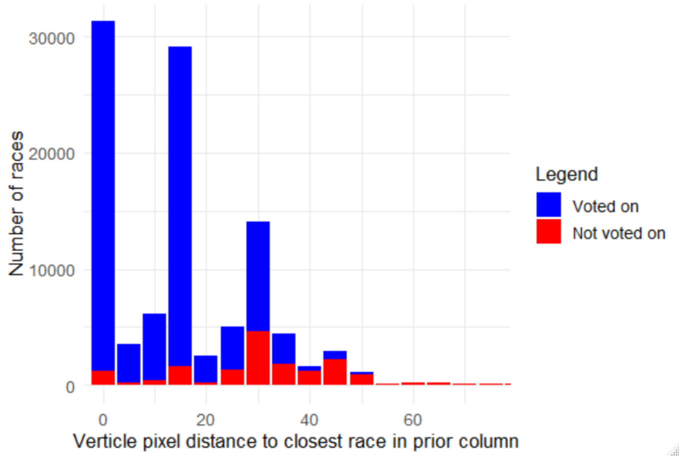


Figure 7: Stacked bar plot of the number of races voted on and not voted on across all model runs, plotted according to the vertical distance between the race and the closest race in the last column (bins of 5 pixels).

This graph more than any other illustrates the model's tendency to miss races that are not lined up in a row; building and running the simulation allows us to identify what these races are for any given ballot.

Conclusion

Races were more likely to be missed if they were smaller, out of alignment with the races in other columns, or more cramped overall. These are all characteristics of bad ballots that our model detected organically. The detection behavior emerged out of the design of the strategy; it was *not* hardcoded. The fact that the model's error behavior was unplanned and emergent is in line with the long-term plan of building models that can produce novel errors on novel ballots.

Notably, using a non-standard macronavigation strategy amplified our ability to detect bad ballots. For instance, a strategy moving in the same direction as the races were originally placed might not mind if the races were very close together, but any other strategy would. Ballot designers need to cater to less common strategies, so an ability to detect when ballots will cause systematic errors in voters using these strategies is crucial.

Indeed, we should note that the average error for this strategy is far higher than the average error for all voters, even assuming as we did that once a voter found a race they would successfully vote on it (choosing a perfect micronavigation strategy, in the parlance of our model). Most real voters probably use a more successful macronavigation strategy. They also may take additional steps we do not yet account for, like scanning the ballot again to see if they missed any races. However, if even a subset of voters uses this strategy, or one like it, then we must account for them in our model, as a subset of voters can still have a deciding impact on a close race.

Thus, one of our next steps will be to map the space of macronavigation strategies by running eye tracking experiments. This research will seek both to find new types of voting strategies and to estimate their prevalence in the voting population. Then, once we build models that represent all of these voting strategies, we will be able to build a ballot analysis tool that runs ballots through each model and weights the resulting error rates by how often people actually use the strategy. Our goal is to be able to use this tool to come up with a global error rate prediction for an arbitrary ballot, preventing badly designed ballots from ever reaching voters.

To implement these new strategies, we will need to expand the capabilities of ACT-R itself. We plan to start by extending the visual grouping module to group objects in a hierarchy and by adding new options for visual navigation. With these new capabilities, we will be able to build new sub-strategies for the model, including new ways for the model to encode the candidate, party, and race groups and new ways for the model to find and click the circle corresponding to a candidate. Each strategy will have a characteristic error pattern like we described in this paper, and together the set of strategies will span the possible space of errors.

Thus, while some of the findings in this paper may seem obvious, they must partly be viewed in the light of the wider project. Our model was able to vote on a wide array of ballots that looked visually different and successfully make consistent errors. More than just characterizing the type of ballots and races that are more disposed to be skipped by a specific voter, these findings confirm the feasibility of attempting to eventually predict errors in novel ballots.

Furthermore, the model makes an interesting additional prediction: since our model is more likely to miss races in the center and right columns, and more likely to miss smaller races, the models predicts that average voter error should be *higher* on down ballot races in the real world (as some voters may use a similar left to right strategy). This skew is likely to be more severe in years with a presidential race, since there are often many candidates running for president, meaning that the first race in the left column would be very long, thus making it more likely that other columns races will not be aligned.

We can even use our results to generate applied advice for a hypothetical election official who must build a ballot with races of varying length. Such an official should strive to line up race headers as much as possible, sacrificing races per page by leaving blank space so that races can be aligned (this would help increase accuracy not only with the specific strategy we tested, but any strategy that goes left to right). Moreover, the official should try not to squeeze races into the bottom right corner, and in general try to keep the ballot uncluttered by putting as much space between races as possible. The official might even consider making the space *within* races more cramped to make the delineations *between* races clearer, although this will introduce the possibility for a voter filling in the wrong bubble or missing the candidate they want to vote for. Future models we build will predict these errors as we continue towards our goal of constructing a model that can simulate all possible voter behavior.

Acknowledgments

This research was supported by grants #IIS-1920513 and #CNS-1550936 from the National Science Foundation. The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies or endorsements, either expressed or implied, of NSF, the U.S. Government, or any other organization.

References

Anderson, J.R. (2007). *How can the human mind occur in the physical universe?* New York: Oxford University Press.

Chisnell, D., & Quesenbery, W. (2018). How a badly-designed ballot might have swayed the election in Florida. Washington Post, November 12, 2018. Retrieved from https://www.washingtonpost.com/outlook/2018/11/12/how-badly-designed-ballot-might-have-swayed-election-florida/

Lindstedt, J. K., & Byrne, M. D. (2018). Simple agglomerative visual grouping for ACT-R. In I. Juvina, J. Houpt, & C. Myers (Eds.), *Proceedings of the 16th International Conference on Cognitive Modeling* (pp. 68–73). Madison, WI: University of Wisconsin.

Norden, L., Kimball, D., Quesenbery, W., & Chen, M. (2008). *Better Ballots*. New York, NY: Brennan Center for Justice, NYU School of Law.

Wand, J. N., Shotts, K. W., Sekhon, J. S., Mebane, W. R., Herron, M. C., & Brady, H. E. (2001). The butterfly did it: The aberrant vote for Buchanan in Palm Beach County, Florida. *American Political Science Review, 95*(4), 793–810.

Wang, X., Lindstedt, J. K., & Byrne, M. D. (2019). The model that knew too much: The interaction between strategy and memory as a source of voting error. In Stewart, T.C. (Ed.) *Proceedings of the 17th International Conference on Cognitive Modeling* (pp. 283–288).Waterloo, Canada: University of Waterloo.