The model that knew too much: The interaction between strategy and memory as a source of voting error

Xianni Wang[1], John K. Lindstedt[1], Michael D. Byrne[1,2]
{xw48, j.k.l, byrne}@rice.edu
[1]Department of Psychological Sciences, [2]Department of Computer Science
6100 Main St., MS-25, Houston, TX 77005 USA

Abstract

This paper presents a family of models of a voting task we developed in order to investigate how errors arise from the interaction between strategy and knowledge. We crossed four task strategies with five different declarative memories and two visual strategies to yield a total of 40 different ACT-R models, and then tested the models through Monte Carlo simulations with 500 runs of each model. The findings suggest that some strategies work best when knowledge is incomplete, and that more task knowledge can lead to more errors in the recall process. These results highlight the importance of studying human error using a thorough exploration of the strategy space.

Keywords: ACT-R; error prediction; voting

Introduction

Human error is important for both theoreticians and practitioners to understand human cognition and performance. While theoreticians like to collect and classify errors, practitioners are more interested in their remediation, prevention, and even elimination. However, research on how to bridge the gap between the theoretical and practical areas is still not particularly common. In this paper, we describe an error prediction method that connects theoretical and practical work on human errors. This method accounts for human performance in routine behaviors using computational modeling and ultimately we hope can be used to predict human error before a system is implemented.

Error prediction methods are often based on traditional hierarchical task models (e.g., Annett & Duncan, 1967), which often assume that the processing system is explicitly hierarchical in structure and therefore break down complex tasks into hierarchies and sub-goals. Botvinick and Plaut (2004) suggested that hierarchical schemas and goals are not always necessary, at least in routine behavior. Instead, they presented a recurrent network model that uses recurrent connections within a network, which map from environmental inputs to action outputs, to represent an everyday task. However, Cooper and Shallice (2006) contrasted this recurrent network model with their more traditional, hierarchically structured interactive activation model. They criticized Botvinick and Plaut's recurrent network approach, describing a set of problems with the approach, such as its behavioral inflexibility, and concluded that hierarchical structures are still necessary and play a causal role in the control of behavior.

Another approach to studying human error is to create human performance models using ACT-R (Anderson, 2007). This goes one step beyond models based on a traditional hierarchical structure by using cognitive architectures. ACT-R is a computational cognitive architecture that simulates and integrates human cognition, attention, and motor behavior. This helps researchers to understand how people organize knowledge and produce behavior in different ways. There are several published ACT-R models that can make the same errors as people (e.g., Anderson, et al., 1998; Halbrügge, Quade, & Engelbrecht, 2015; Lebière, Anderson, & Reder, 1994; Trafton, Altmann, & Ratwani, 2011).

However, it is not easy to predict human error using ACT-R. First, there are many types of errors, but a human performance model usually only makes a specific kind of error. If there is a complex working system that contains several sub-tasks, it will take time and effort to create models that cover all possible errors. Second, in general, computational human performance models are fitted to and/or make predictions about average human behavior. However, predicting errors cannot simply be a question of fitting the mean, because even if the average person does not make an error, there may still be a substantial number who do.

Our domain for error modeling is voting. People usually think that filling out a ballot is a simple task, but, in reality, unintentional undervotes, overvotes, or votes for the wrong candidate are very common in almost all elections. An undervote occurs when the number of votes is less than the maximum number allowed in a race, and an overvote occurs when the number of votes is more than the maximum number allowed. One reason for all these errors lies in the poor designs of the ballots, which fail to support human perceptual and cognitive limitations. There is clear evidence that ballot design problems have affected the outcomes of multiple elections in the United States (Laskowski et al., 2004). For example, more than 2,000 votes intended for Gore were cast for Buchanan in Palm Beach County, Florida, during the 2000 elections due to the use of the infamous butterfly ballot (Wand et al., 2001).

A standard usability evaluation prior to deployment would likely detect poor designs and prevent errors. However, usability specialists are rarely asked to perform such tests prior to an election. Instead, election officials, who have little formal training or the expertise in assessing ballots, are left to the task. In addition, most elections in the U.S. are administered at the county level, and there are over 3,000 counties in the U.S. Within each county, there are often hundreds of different precincts, each with a slightly different ballot style, meaning that, for each national election, tens of thousands of ballot designs are deployed. This makes conducting a traditional usability test for every single ballot intractable due to the problem scale.

While it is impossible to perform usability testing on every ballot before every election deployment, some initial work has been done on predicting errors in voting tasks. In Greene (2010), an ACT-R model was presented that could sometimes make the same mistake that voters made in Sarasota, Florida in 2006. In this case, the first DRE screen contained one race, but there were two races presented on the second screen. This layout inconsistency led to 13.9% of votes being undervotes in the top race on the second screen. Greene modeled these first two screens to explore two voting strategies. The first strategy was to read the first screen from top to bottom before selecting a candidate, and then recall a useful location from the first screen to use to direct the visual search on the next screen. With this strategy, the model used the first screen to set expectations about where to find relevant landmarks (e.g., titles of races); it could then miss the critical top race on the second screen when the model extended those expectations from one screen to the next. The second strategy was to read both screens from top to bottom, without any recall. In contrast to the first strategy, the second strategy did not result in a critical top race undervote.

Greene's (2010) model offers a meaningful opportunity for computational human performance modeling to make a unique contribution to the voting field. However, this model does not reflect the full complexity of voting. Different voters almost certainly approach ballots differently. It is therefore critical that the models reflect not just one or two voting strategies, but the entire range of behaviors, so that specific interactions between voting strategies and ballot designs can be uncovered.

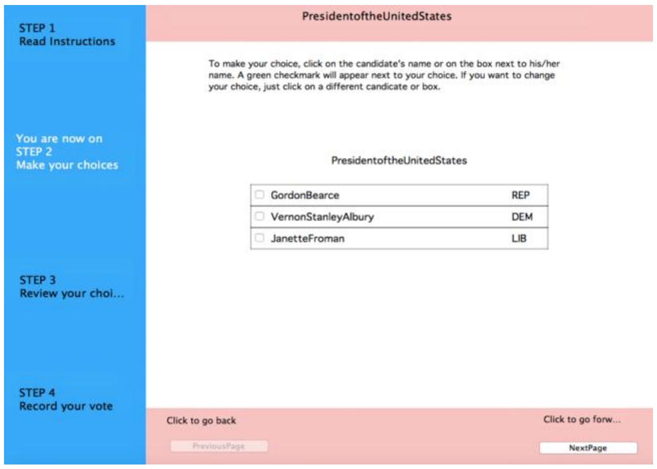
To capture more of the voting complexity, we developed a model-based approach that covers a family of voting strategies using ACT-R. For each model, the memory strategy, ballot knowledge, and visual search strategy were considered independently. Memory strategy represents how voters access their memories when they cast a vote; ballot knowledge defines voters' level of knowledge of the races and candidates; and visual search strategy indicates voters' visual directions when conducting a visual search. In total, our system consists of 40 voting models that crossed four memory strategies with five kinds of ballot knowledge and two visual strategies.

When it comes to visual search, humans have a remarkable ability to organize their perceptual inputs. The human visual system tends to group individual items in a visual image into larger structures under certain circumstances. This allows for the more efficient use of attention but sometimes leads to critical errors in executing a task. For example, the ballot used in Wisconsin in 2002 led to many unintentional votes. On this ballot, the race for governor was split across two columns, which led many voters to consider the two visual groups as two races. Many voters voted twice, once in each column. To handle situations like this, our system makes use of a visual grouping algorithm that enables more realistic visual scanning behaviors (Lindstedt & Byrne, 2018).

In short, our model-based system assessed a ballot layout with a family of voting models. Each voting model was tested multiple times, and the average across those runs was calculated. After running every ballot through each model repeatedly, all combinations of strategies and knowledge that generated high error rates were identified.

The Voting Task

Our system was implemented for an emulated voting task using the VoteBox task environment. Multiple experiments have been published in which human subjects voted using VoteBox (e.g., Everett, 2007; Everett, et al., 2008). This voting task contains 21 races that share a consistent layout (see Figure 1). The layout was designed to be easy to understand, with a relatively simple display that comprised the voting instructions, title of the race, candidates' names and party affiliations, a "previous page" button, and a "next page" button, all clearly arranged and presented across the screen.



PresidentoftheUnitedStates

To make your choice, click on the candidate's name or on the box next to his/her name. A green checkmark will appear next to your choice. If you want to change your choice, just click on a different candicate or box.

PresidentoftheUnitedStates

☐ GordonBearce	REP
☐ VernonStanleyAlbury	DEM
☐ JanetteFroman	LIB

Figure 1: Mock ballot of a presidential race.

All versions of the model contain two phases. The first is a studying phase in which the model studies the display thoroughly to retain group information produced by the visual grouping algorithm. The second phase is a voting phase; after obtaining and storing group information during the first phase, the model now has expectations about where to look. It directs its gaze to the appropriate place and then makes a vote.

Modeling Strategies

A total of 40 voting models were developed. Each model includes a memory strategy, ballot knowledge, and a visual search strategy. To produce a comprehensive error prediction, multiple plausible versions were considered for each component. The details of each component are described in the sections below. We defined four memory strategies, five kinds of ballot knowledge, and two visual search strategies.

Memory Strategies

Voters have to remember their choices, and they access their memories in different ways. There are two primary memory strategies for simple form-filling tasks like voting: retrieval and recognition. Some voters can simply recall the names of those for whom they intend to vote, at least in some races. For example, many voters, when prompted, can retrieve from memory the candidate for whom they intend to vote in presidential elections. Other voters may instead scan the list of names first to try to recognize their preferred candidates. Some voters vote almost exclusively according to party affiliation but then have to remember which races, if any, have exceptions. Some voters may rely on party affiliation if they can neither recall for whom they intended to vote nor recognize any of the candidates' names on the list. While some voters may also write out a list and bring it into the voting booth, it is not clear how common this is, and it is, in fact, illegal in some jurisdictions. So, we did not consider this strategy.

Our models capture four memory strategies one could reasonably expect a voter to employ—a strictly retrieval-based strategy, a strictly recognition-based strategy, a retrieval-then-recognition contingency strategy, and a simple party-only look-up strategy (in case of exceptions to their default party). The first strategy represents the scenario in which the model first tries to retrieve the candidate's name from memory. If the model fails to recall the name, then it relies simply on a party affiliation. The second strategy considers the situations in which the model first tries to retrieve their choice, but, if the retrieval fails, it then scans the list of names and votes for the one it recognize. If recognition also fails, it votes by party affiliation. For the third strategy, the model does not even attempt to retrieve; rather, it scans the list of names to see if it can recognize any of them. If recognition fails, it votes by party. For the last strategy, the model simply votes based on party affiliation. It first retrieves the specific party affiliation for specific races, but, if the retrieval fails, default party affiliation becomes the only criterion. The last step of these four memory strategies—voting by default party affiliation—is used only when all the previous steps fail. The default party affiliation could be either the Democratic, Republican, or Libertarian Party.

Other memory strategies are certainly possible, but it is unclear how a voter could use the contents of their memory to vote in a meaningfully different manner without substantial overlap with one of the strategies listed above.

Ballot Knowledge

Voters have different levels of knowledge about the races and candidates. Some voters might have encoded all of the candidates' names, some may only know the names of candidates they intend to vote for, and some may only have parts of the intended candidates' names in their memories. In addition, ballot knowledge is not always easy to recall. Some voters may only remember their choices for the first few races because it is much more likely that voters will have more frequent exposure to top-of-the-ballot candidates. ACT-R represents situations like this using base-level activation, which reflects the recency and frequency of a specific memory.

Table 1: Ballot Knowledge

Ballot Knowledge	Candidates' Names	Activations for Intended Candidates
FULL-MEMORY	All candidates	Races 1 to 7: 0.7 Races 8 to 14: 0.6 Races 15 to 21: 0.5
ALL-ROLLOFF	Intended candidates only	Races 1 to 7: 0.7 Races 8 to 14: 0.6 Races 15 to 21: 0.5
ALL-PERFECT	Intended candidates only	All races: 0.8
MOST-ROLLOFF	70% of intended candidates	Races 1 to 3: 0.8 Races 4 to 7: 0.7 Races 8 to 11: 0.6 Races 12 to 15: 0.5 Race 16 to 21: Abstained
MOST-PERFECT	70% of intended candidates	Race 1 to 15: 0.8 Race 16 to 21: Abstained

Five ballot knowledge types were therefore created (see Table 1). First, we defined three levels of how many candidates' names were stored. The models could remember all candidates' names, only the intended candidates' names, or only the first 70% of the intended candidates' names. Then, we assigned two types of activations for intended candidates: roll-off activations and constant high-level activations. Models with roll-off activations are most familiar with the candidates for the first several races; then, as they progress down the ballot, their familiarity with candidates decreases. In the second condition—constant high-level activations—the models are highly familiar with all races to the same degree. Note that the various contents and activation levels of memory were not chosen as an exhaustive search of all possible knowledge held by voters, but rather as an illustrative sample of common voter scenarios—some voters have certainly done their homework extensively, while others have likely only decided “important” races.

Visual Search Strategies

While reading in a serial order is the most common search strategy, eye-tracking studies have demonstrated that it is not universal (Aaltonen, Hyrskykari, & Räihä, 1998; Fleetwood & Byrne, 2006). People scan displays in different ways: some readers read in a serial, item-by-item pattern, from one corner to its diagonal opposite; some people scan globally and read all the bold, large, or colored headers first; and some simply prefer to scan randomly.

Two visual search strategies were used when looking for candidates: a serial search and a random search. The serial search strategy is a serial item-by-item search with a left-to-right, top-to-bottom pattern. With the random search strategy, the models conduct a random search.

Model Evaluation

Method

The first issue to address is the number of Monte Carlo replications. We used the approach outlined in Byrne (2013) based on confidence intervals. We expected the overall error rate generated by the model to be around 5% and wanted the 95% confidence intervals for the model predictions to be no wider than 2% in either direction. The table in Byrne (2013) shows this requires 457 model runs; we ran 500 per model to be slightly more conservative.

Error Predictions

For each model run, the ballot, as completed by the model, was compared with the “intent” initialized at the beginning of the run, and any discrepancies were noted as errors. Errors occurred across the entire voting process. The model might have retrieved an unintended name, recognized an unintended name, or failed to retrieve and then recognized an unintended name. For the model that simply made votes based on party affiliation, it may have retrieved an unintended party. The model may even have failed to retrieve and/or recognize an intended name, and then have voted by default party affiliation. We used Democratic as the default party affiliation for this model evaluation; however, intended candidates’ party affiliations did not always match the default party affiliation. The model occasionally also mis-clicked on the name above or below the intended name.

Overall, our models generated an average 5% error rate across all voting models. This is somewhat higher than

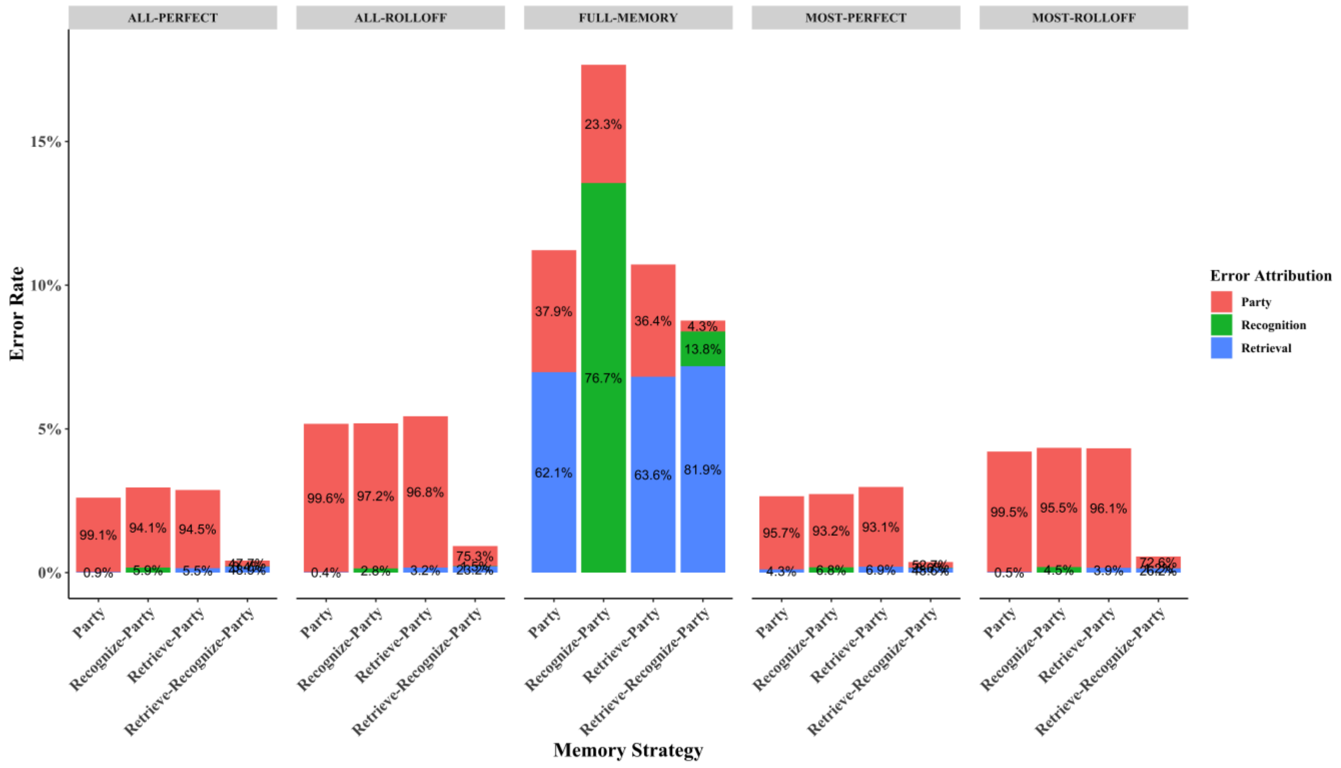


Figure 2: Interaction between memory strategy and ballot knowledge in voting error rates. The bars show voting errors. The five ballot knowledge types are written along the top; each shows the error rates for the four memory strategies. The three colors of the bars indicate the three kinds of processes the model used when it made an error. Red, green, and blue represent the party affiliation, recognition, and retrieval, respectively.

observed human error rates of around 1.5% (e.g., Everett, et al. 2008) and unfortunately it is impossible to compare our models directly to human data since we cannot know the strategies used by people in such studies. Instead, we are interested in how the model strategies interact to produce errors.

Differences in error rates with visual strategies were not found, which means that using either a serial or a random scanning pattern did not affect the voting results. The main story here is therefore about memory strategy and ballot knowledge. We observed differences in voting errors based on the interaction between voting strategy and ballot knowledge.

Figure 2 presents five groups of bars that represent the error rates of the five kinds of ballot knowledge. For each type of ballot knowledge, the percentages of the errors for the four memory strategies are displayed. For the FULL-MEMORY condition, the model generated 9% more errors than the other four ballot knowledge types. The model also generated more errors with roll-off activations for intended candidates. For the MOST-ROLLOFF and ALL-ROLLOFF conditions, the voting model was 2% more likely to make errors than with the MOST-PERFECT and ALL-PERFECT conditions. Additionally, for these four levels of ballot knowledge other than FULL-MEMORY, there were clearly fewer errors with the three-step "retrieve-recognize-party" memory strategy.

We then investigated the error attributions for each vote to determine which process the model was using when it made an error. In Figure 2, each bar is partitioned into three colors, which represent the three kinds of process the model was using when it made an error: retrieval, recognition, or party affiliation. For the FULL-MEMORY condition, most of the errors occurred in the recognition and/or retrieval processes. Within FULL-MEMORY, 7% more voting errors were generated with the "recognize-party" memory strategy. However, for the other four kinds of ballot knowledge, differences in error attributions with memory strategies were not apparent; most of the errors were generated in the last steps—voting by party affiliation.

Discussion

The error predictions indicate that extra ballot knowledge actually led to more errors, especially with the involvement of recognition. Common sense would suggest that a broader knowledge base should help to mitigate mistakes, but this is not always the case. Another example that also suggests that a strategy works best when knowledge is incomplete is the recognition heuristic. The recognition heuristic describes a situation where, if one of two objects is recognized and the other is not, the recognized object is more likely to be selected (Goldstein & Gigerenzer, 2002). This strategy requires ignorance to make a choice—if people know everything or nothing about the options, it simply does not work. For example, for the question "which city has a larger population?" most people choose Dublin over Nenagh since they can recognize Dublin only, but it is harder for people to make a selection if the choices become San Diego and San Antonio, as they are more likely to recognize both of these cities. Similarly, in the voting task in our study, the models knew everything in the FULL-MEMORY condition, including both the intended and unintended candidates' names. Thus, compared to the other four ballot knowledge types, the memory strategies did not work well with FULL-MEMORY, and more errors occurred in the recognition processes.

Because of the more frequent recognition errors, one thing we can expect with the FULL-MEMORY condition is a greater impact of candidate name order. Voters who cannot recall their intended candidate's name must scan the list of names and see if they can recognize any, and their choices can be biased by the order in which candidates' names appear on the ballot (Miller & Krosnick, 1998). In our study, the model with "recognize-party" memory strategy checks each candidate, sees if it recognizes, and if so, votes for it. Since some voters use top-to-bottom visual search, an advantage for the top candidate can be predicted.

Another finding has to do with the interaction between task knowledge and recall performance. Schooler and Anderson (1997) suggested an association between the number of choices and recall performance, positing that the more choices we have, the more likely we are to make a recall error at each name. We observed the same relationship in our models. The FULL-MEMORY condition contains both intended names and unintended names, and the models could either retrieve an intended name or an unintended name for each race in that condition. It was therefore more likely to make errors in the retrieval process since incorrect answers are available. However, with the other four ballot knowledge types, there are only intended names available in memory. Wrong names were therefore less likely to be retrieved with these four levels of knowledge.

We can also conclude from the error predictions that the three-step "retrieve-recognize-party" memory strategy had a better performance than the two-step memory strategies. As can be seen in Figure 2, a large portion of the errors came from the last steps, voting by party affiliation, across five levels of knowledge. Comparing to the two-step strategies, the additional one step prevented errors that could be made in the last step, and so the least amount of errors was generated with the three-step memory strategy.

Note that the errors made here are not the result of poor ballot design. However, we believe that further interactions, those between strategy, knowledge, and ballot design, will show how the visual layout of the ballot can influence error rates. Poor layouts may not induce all voters into error, but differentially affect those who use particular strategies. Furthermore, we believe that these kinds of errors are not limited only to filling out ballots, but likely occur in other tasks that are essentially form fill-in, such as interacting with electronic health records.

Our model-based system represents the first use of ACT-R as an error prediction tool to diagnose if there are particular combinations of strategies that lead to error. The idea that one can understand the error space by modeling only one strategy or predicting mean behavior is likely to miss critical combinations of factors that produce errors. Our results demonstrate that subtle interactions between strategy and knowledge can have substantial effects on error rates. Thus, it is critical to consider multiple combinations of both when attempting to model errors, even in a task that appear as simple as voting.

Acknowledgments

This research was supported by grant #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

Aaltonen, A., Hyrskykari, A., & Räihä, K. J. (1998). 101 spots, or how do users read menus? In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 132-139). ACM Press/Addison-Wesley Publishing Co.

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

Anderson, J. R., Bothell, D., Lebiere, C., & Matessa, M. (1998). An integrated theory of list memory. *Journal of Memory and Language*, *38*(4), 341-380.

Annett, J., Duncan, K.D., 1967. Task analysis and training design. *Journal of Occupational Psychology 41*, 211–221.

Botvinick, M., & Plaut, D. C. (2004). Doing without schema hierarchies: a recurrent connectionist approach to normal and impaired routine sequential action. *Psychological Review*, *111*(2), 395-429.

Byrne, M. D. (2013). How many times should a stochastic model be run? An approach based on confidence intervals. In *Proceedings of the 12th International Conference on Cognitive Modeling* (pp. 445-450). Ottawa: Carleton University.

Cooper, R. P., & Shallice, T. (2006). Hierarchical schemas and goals in the control of sequential behavior. *Psychological Review, 113*(4), 887-916.

Everett, S. P. (2007). *The usability of electronic voting machines and how votes can be changed without detection* (Doctoral dissertation). Rice University, Houston, TX.

Everett, S. P., Greene, K. K., Byrne, M. D., Wallach, D. S., Derr, K., Sandler, D., & Torous, T. (2008). Electronic voting machines versus traditional methods: Improved preference, similar performance. *Human Factors in Computing Systems: Proceedings of CHI 2008* (pp. 883-892). New York: ACM.

Fleetwood, M. D., & Byrne, M. D. (2006). Modeling the visual search of displays: A revised ACT-R model of icon search based on eye-tracking data. *Human-Computer Interaction*, *21*(2), 153-197.

Goldstein, D. G., & Gigerenzer, G. (2002). Models of ecological rationality: The recognition heuristic. *Psychological Review*, *109*(1), 75.

Greene, K. K. (2010). *Effects of Multiple Races and Header Highlighting on Undervotes in the 2006 Sarasota General Election: A Usability Study and Cognitive Modeling Assessment* (Doctoral dissertation). Rice University, Houston, TX.

Halbrügge, M., Quade, M., & Engelbrecht, K. P. (2015). A predictive model of human error based on user interface development models and a cognitive architecture. In *Proceedings of the 13th International Conference on Cognitive Modeling* (pp. 238-243). University of Groningen, Groningen, the Netherlands.

Laskowski, S. J., Autry, M., Cugini, J., Killam, W., & Yen, J. (2004). *Improving the usability and accessibility of voting systems and products*. NIST Special Publication 500-256. Retrieved from https://user-centereddesign.com/files/NISTHFReport.pdf.

Lebière, C., Anderson, J. R., & Reder, L. M. (1994). Error modeling in the ACT-R production system. In *Proceedings of the Sixteenth Annual Conference of the Cognitive Science Society* (pp. 555-559). Erlbaum Hillsdale, NJ.

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.

Miller, J. M., & Krosnick, J. A. (1998). The impact of candidate name order on election outcomes. *Public Opinion Quarterly*,*62*(3), 291-330.

Schooler, L. J., & Anderson, J. R. (1997). The role of process in the rational analysis of memory. *Cognitive Psychology*, *32*(3), 219-250.

Trafton, J. G., Altmann, E. M., & Ratwani, R. M. (2011). A memory for goals model of sequence errors. *Cognitive Systems Research*, *12*(2), 134-143.

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.