Efficient Data Collection for Upcoming Offices via LLMs

Sophia Mirrashidi[1], Luke Dillon[1], Wentao Zhang[1], Candis Wu[1], Colin Nguyen[1], Pramukh Shankar[1], Yan Lashchev[2,3]1

University of California, Santa Barbara; [2] NationBuilder [3]

Who is NationBuilder?

NationBuilder specializes in content management and CRM software originally developed for political campaigns and nonprofits. Over time, it has expanded to serve a wider range of clients across various sectors. The platform is designed for rapid campaign deployment, enabling users to quickly build websites, create databases, and launch targeted communications. It also features powerful data analytics and micro-donation capabilities, allowing organizations to track supporter engagement, streamline fundraising, and gain actionable insights with speed and precision.

Introduction

Run for Office

Run for Office

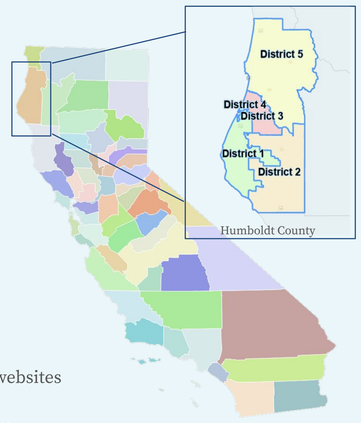
According to Ballotpedia, 70% of elections in 2024 went uncontested. Run for Office, a tool hosted by NationBuilder, is designed to help lower that number by enabling citizens to identify seats up for election in the coming years. Historically, Run for Office relied on volunteers manually sifting through PDFs and websites for each jurisdiction, a process that is labor-intensive and prone to delays and errors. Our goal is to automate the process of finding data regarding upcoming elections in the United States.

Problem:

- Requires volunteers
- Slow and tedious
- Delays and errors

Solution:

- Automate the search and retrieval of documents
- Relevancy Filter
- Data Extraction



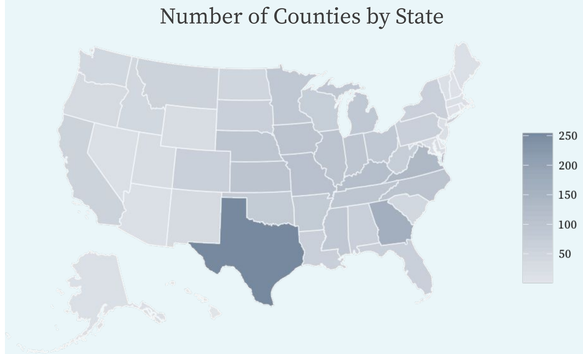
Challenges:

- Heterogeneous data
- Old and antiquated websites (Not SEO optimized)
- Varying PDF structure.

We started by experimenting with data from Humboldt County because their PDFs were the most comprehensive and easy to find. Their election websites were also easy to navigate, making it an ideal trial run. After fine-tuning our pipeline on this country, we eventually scaled up to include all 58 counties in California.

Crawling for Data

The National Voter Registration Act of 1993 mandates that each state designate a chief election official, often the Secretary of State. These officials oversee federal and statewide election data, while city and county officials manage local information, typically published on their respective clerk's websites. The data is often in PDF or HTML format located somewhere on the official election website.



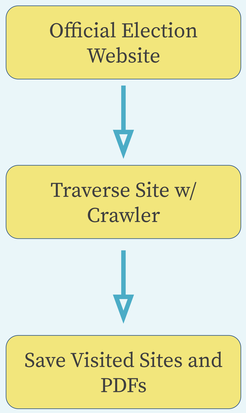
To find this data, we built a crawler using the Playwright package in Python. The crawler works from a predefined list of California county election websites that was gathered from the California Secretary of State website. Once the crawler has accessed a website, it traverses every link that contains a set of election-related keywords and saves each visited link for further processing.

The crawler was run on all of counties in California, saving approximately 1,500 pieces of unprocessed data.

Since the crawler saves every link that it visits, much of this data is irrelevant to our goal.

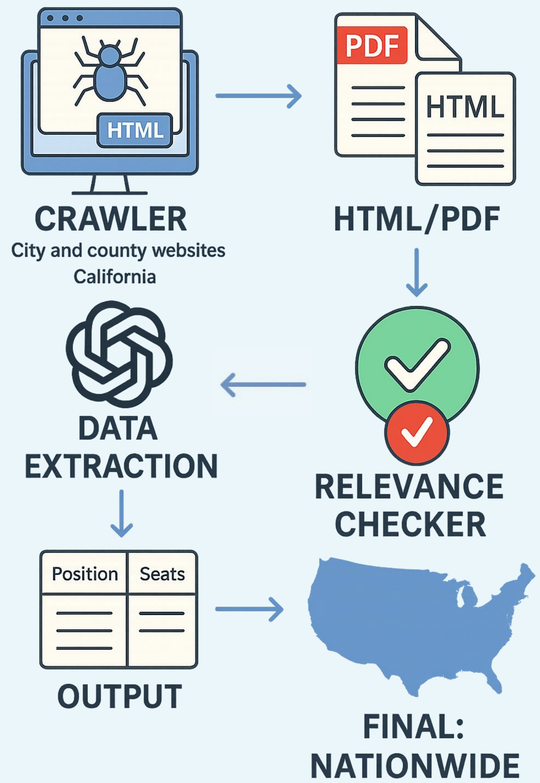
To scale this step, one only needs to add more official election websites to crawl.

The next step in the pipeline is process both the relevant and irrelevant data.



Processing Relevant Data

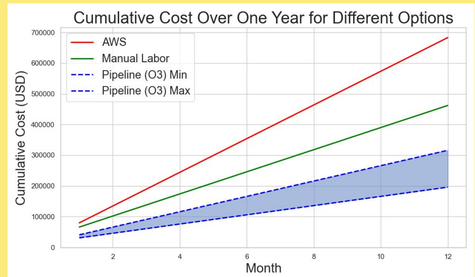
The crawler saves all of the link it visit, which includes hundreds of PDF files and HTML pages from official election websites. The gathered files are first sent to a "relevance checker". We use OpenAI's gpt-4.1 to check if whether or not each PDF contains the data that we are looking for. This is to help minimize the costs by ensuring that we only process PDFS that are deemed relevant. The filtered PDFs are then sent to OpenAI's o3 model for processing. From each file, we want to extract data such as position titles, term lengths, available seats, and signature requirements. In our testing, we found that this reasoning model is able to read through long, complicated PDFs and extract the data with near perfect accuracy. The data is saved as a JSON file and then will be filled into a database for Run for Office to pull from.



This marks the complete pipeline. We start at the official election sites, save the site HTML and linked PDFs and send them to the relevancy checker. The files that make it through the checker are sent to o3 for data extraction, and that data is saved.

Summary of findings

- **Competitors' Approach:**
  - Ballotpedia and Lambert Labs use AWS Organizations for web crawling, costing around $54,000/month to crawl and store election data.
- **Manual Web Crawling:**
  - Estimated cost of $36,000/month for manual data collection.
- **Our Solution:**
  - $16,000/month to gather election data for the entire country.
  - Significantly more cost-effective than both manual web crawling and using AWS.



Future work

The direct next step would be to connect the end of our pipeline to the Run for Office database so that the website can be populated with our collected data.

Now that we have completed this proof of concept for California, the next step would be to expand the pipeline to include the other states and their respective counties.

Beyond that, we could improve the tool by integrating scheduled updates to capture changes to county election websites over time in case there are any changes to the website format.

Lastly, depending on what tools are available in the future, we can transition to cheaper models, to ensure equal access in running for office for all.

References and Acknowledgments

[1] Ahluwalia, A. & Wani, S. (2024). Leveraging Large Language Models for Web Scraping. arXiv, 2406.08246.

[2] Krosnick, R. & Oney, S. (2023). Promises and Pitfalls of Using LLMs for Scraping Web UIs. CHI Workshop on Computational Approaches for User Interfaces, 2023.

A special thanks to NationBuilder and our wonderful advisor Yan Lashchev for their guidance!