Revision Memo

The manuscript we have submitted is a revision of PA-2018-124, which was rejected. We have revised it from an article to a letter, in accordance with the reviewers' and editor's recommendation. We thank the editor for recommending that we revise our manuscript and resubmit it as a letter for consideration at *Political Analysis*. Throughout the revision process, we have incorporated the feedback of the reviewers. In this memo we have separated the editor's and reviewers' comments into separate points. Under each point, we describe how we have revised the manuscript in response to the feedback provided. We generally agree with the criticisms offered, and think that the manuscript has improved substantially as a result of incorporating this feedback.

Editor

E.1: *Having said all of that, I would welcome a de novo contribution in the form of a letter (per our guidelines) with the specific software details furnished in an appendix to be placed online if published (or preferably as an R package automating some of the decisions).*

Addressed: We have revised the manuscript in exactly this format—a letter with an associated R package that wraps and automates most of the pipeline we describe and use in the paper.

E.2: *Of course this letter would have to be written such that there is a methodological contribution independent of the appendix.*

Addressed: The methods contributions offered in the letter are four-fold. First, the construction and description of the step-by-step pipeline required to go from raw website files to clean substantive text blocks. Second, we introduce the boilerpipe algorithm to the field of political science—a method that will be of high value to anyone using websites as sources of textual data. Third, we provide the R package. Fourth, we provide a new dataset of clean text extracted from city websites.

Reviewer 1

R1.1: *If pipeline is a research output, it should be made available (e.g. in form of an R-package)*

Addressed: We have implemented and wrapped the core components of our pipeline in an R package, gov2text. The package will be publicly distributed on GitHub upon acceptance for publication. In Table 1 we present the methods and algorithms that are wrapped in the R package.

R1.2: *I agree that extraction of plain text (body text extraction) is almost always the problem in online data collection, but author resort to manual coding and supervised machine learning to do the task. Since manual coding is not always the option, the*

authors should have tried to develop semisupervised or unsupervised method. Creation of such a tool is certanly a great contribution.

Addressed: We now introduce the boilerpipe algorithm for this purpose, which does not require the development of a corpus-specific classifier.

R1.3: *I think the use of Selenium is appropriate because it replicates humans accessing website. However, I do not understand why the author used wget to download pages. There must be dynamic elements in the pages that are generated or inserted by javascript. I consider this is a deficiency of the pipeline, which leading to incomplete data collection.*

Addressed: We have added a discussion of this point to the conclusion—in response to this comment and to R3.3, where Reviewer 3 asks about limitations of our methods. Our objective is to provide a scalable pipeline. While we agree that we could use Selenium to gather the dynamic content on any given site, we would need to customize the process for each website, whereas wget can be applied across sites without site-specific supervision. We acknowledge that this is a limitation of our process—perhaps the biggest limitation, and note that it is an open area for future research. Furthermore, we investigated whether the presence of JavaScript on a site is associated with the amount of textual content we scrape from the site. We calculated the correlation between the number of <script> HTML tags on a city's site and the number of text tokens scraped from that site. The correlation is -0.059, which indicates a weak relationship between the use of JavaScript and the amount of content we scrape. We reported on this in Footnote 6.

R1.4: *Also, there is no need to use Selenium only to handle redirections.*

Addressed: We explored the possibility of using Selenium as our general-purpose scraper for this project. We developed a proof-of-concept Selenium-based webcrawler in response to R1.4, but realized that in practice, a browser-based scraper that recursively downloads websites still falls short of the necessary functionality---in particular the ability to function fairly consistently across sites. Wget is a robust tool with the capability to handle errors and deal with unstable network connections (on either end) appropriately. It possesses a number of important features, such as the ability to recover failed or partial downloads, and timestamping. By contrast, we found that Selenium was not designed for our purpose of deploying a scraper that would work fairly consistently across many sites. For example, pop-ups can obscure a website from Selenium, and have to be dealt with for each website individually. Consequently, wget is a superior tool when it comes to reliably downloading textual content. Developing a Selenium-based alternative would require custom development for each city website.

R1.4: *The authors removed boilerplate expressions line byline after converting files into plain texts, but I am not sure how line byline elimination is effective, because inline elements (e.g. span tags) can occur in the same line as substantive content. They should have considered the nested structure of HTML documents for effective body text*

extraction.

Addressed: Since submitting the first version of the paper, we discovered the boilerpipe algorithm, which is very effective at extracting substantive blocks of text from websites (and has been tested and verified in published work). We abandoned our initial line-by-line approach in favor of using boilerpipe. Of course, we did not create boilerpipe. However, from what we can tell, it has not yet been used in political science (which we note in the paper), and we do ease its application to government website files through its incorporation in the gov2text package.

R1.5: *It is strange to remove tokens before applying POStagger (spaCy), because accurate lemmatization requires syntactical parsing of the original texts.*

Addressed: We agree with this criticism, and have now moved lemmatization up in the processing pipeline, as is made clear in Table 3.

Reviewer 2

R2.1: *I think the manuscript would be better suited for PA as a letter focused on the toolchain and explicitly aim at a broader audience than researchers working with local government documents. Revised into that form, it would work better if it would give more information on why and how the various components of the toolchain are preferable over the alternatives, and what are their limitations.*

Addressed: We have revised the paper to be formatted as a letter focused on the toolchain. Our intention is to focus on government websites in general, with the running application of city websites. We removed much of the theoretical discussion of local government websites, and retain just enough to make sense of the running application. Since there are 8-10 separate tools in the toolchain, we do not have room to engage each one with a detailed comparison of alternatives. However, one improvement along these lines is that we now use a well established (and published) method for website boilerplate removal, so we can at least reference previous work regarding the effectiveness of that methodology.

R2.2: *Perhaps the most important instance are the various methods for boilerplate identification, currently relegated to footnote 11. Here, it would help the manuscript's case a good deal if it could show how boilerplate removal matters for substantive conclusions, and show this not only for the analyzed corpus but also for some published findings.*

Addressed: We agree that it would significantly strengthen our paper to include analysis that illustrates the improvement offered through boilerplate removal. We have included two analyses to respond to this point. The first is a single case, presented in Figure 2, in which we show how boilerpipe is effective at extracting just the substantive text, leaving out all of the menus and other header and footer content. The second is a replication of our main STM results using data processed with a minimal HTML parser.

We present these results in the online appendix. The key illustrative result seen in this table is that two of the most partisan topics are HTML boilerplate topics. No such topics arise in the main analysis using boilerpipe-processed content. In terms of illustrating boilerpipe on other corpora, now that we have converted this paper to a letter, we see this as beyond the scope of our paper.

Reviewer 3

R3.1: *This article is a useful set of practices to help create a datasource for local politics. However, the way that it's written right now it seems more appropriate for a note than a full article.*

Addressed: We have reformatted the article as a letter.

R3.2: *Gathering data in local politics is difficult and recent efforts such as the LEAP project, and Sumner, Farris and Holman's crowdsourcing method aim to make that process more access to scholars studying local and state politics. This website method could supplement these efforts and help scholars in the US (and abroad) study processes at the level where most people interact with the state – at the local level.*

Addressed: We have added a sentence noting that our dataset represents a contribution to the growing literature focused on providing cross-municipality data on local governments (referencing the LEAP project and Sumner et al. as examples). Upon acceptance, we will publicly distribute a clean version of our dataset that includes the substantive text blocks as files, organized by city (i.e., the form of the dataset following Step 4 in the pipeline, as summarized in Table 3). This dataset will be around 4GB. We are more than happy to share the entire dataset of raw website files, but sharing those will require individual cooperation with the recipient since the archive of raw files is so large.

R3.3: *The authors talk about the improvements over other methods that they are offering, but are there drawbacks as well?*

Addressed: The biggest limitation in our methodology is the reliance on wget to collect content from government websites. This method misses dynamic content displayed via JavaScript (i.e., a table populated by querying a database). We added a paragraph to the conclusion in which we discuss this limitation, and call for it to be addressed in future research.

R3.4: *The topic modeling has no valence attached, so while the modeling can speak to broad categories that are being mentioned, can it also enable scholars to say something about credit claiming or blame attribution? Can we say something about the absence of terms across different types of municipalities? Right now, the article uses the partisanship of the mayor as a way to test whether websites across municipalities differ. This doesn't seem to be the most useful example since most offices that people vote for in the US are nonpartisan and the data here can speak to a great deal of other things*

that may define information cities share with citizens and the broader public.

Addressed: We responded to this comment through a couple of edits. First, on the issue of credit-claiming, we pointed out that the greater prevalence of finance-related topics in cities with Democratic mayors could be a reflection of their greater willingness to discuss and take credit for city initiatives to raise and spend money, as is reflected in the new running example from Gary, IN. Second, we added a note highlighting that in 53 of the 60 topics estimated, the 95% credible interval for the effect of mayoral partisanship did not include zero. We acknowledge that most mayoral races are nominally non-partisan, but our results provide strong evidence that the topics conveyed on city websites vary systematically with mayoral partisanship, even controlling for population and income.

R3.5: *If the authors want to stay with the local politics example, it would be helpful to have a running example of a city throughout the text to show each step along the way.*

Addressed: We have added an example involving a message from the Mayor of Gary, IN that is depicted on the city's homepage. The mayor presents a very clear policy message on the homepage, which we depict in a screenshot. We then show how a naive text extraction from that site would include mostly menu items and links to other parts of the website (i.e., boilerplate). Finally, we show how, using the boilerplate elimination method we suggest isolates the Mayor's message---the only substantive text on the page.

R3.6: *In addition, the topic model technique should allow the authors to say something about the topics that occur in websites along multiple dimensions that they already have measures of like city size, region, median income.*

Addressed: We included median income and population in the STM. We do not have the room in the letter to include presentation and discussion of these results in the main text. However, we have updated the online appendix to include visualization and brief discussions of these results. The results are intuitive. We find that the websites of wealthier cities disproportionately contain topics related to initiatives that go beyond basic municipal services (e.g., downtown revitalization, wildlife conservation), and the websites of larger cities disproportionately contain topics focused on issues that are commonly associated with large cities (e.g., crime, homelessness, diversity).

R3.7: *It would also be helpful to have a table or a figure that lays out the steps from preprocessing to topic modeling along with all of the R packages so that it's more of a user guide for readers.*

Addressed: We have added a table to the paper (Table 3) in which we list the steps in the pipeline, the software on which each step depends, and indicate whether the step is incorporated into our R package, gov2text.