Final Report: A Replication of *Measuring and Explaining Political Sophistication through Textual Complexity* (Benoit et al., 2019)

PPOL 6801: Text as Data: Computational Linguistics

Maria Bartlett & Wendy Shi

February 18, 2025

I. Introduction

Various metrics for textual complexity have emerged from education and psychology research, most notably the Flesch Reading Ease (FRE) score (Benoit et al., 2019). However, Benoit et al. (2019) note that a textual sophistication measure specific to political texts has not previously existed. In their 2019 journal article, *Measuring and Explaining Political Sophistication through Textual Complexity*, Benoit, Munger, and Spirling seek to explore key determinants of complexity for political texts and subsequently develop a widely applicable political text sophistication model that remedies several shortcomings of FRE-type models.

Benoit et al. (2019) argue that a complexity model tuned on political text is important to help answer questions of whether political text complexity has evolved over time and the effects of such changes. While FRE and similar text complexity models have commonly been applied to political texts, Benoit et al. (2019) highlight several domain and statistical reasons why this practice should be challenged. First, the FRE is an older metric not specific to political texts: it was designed and refined between the mid-20th century and mid-1970s in the context of U.S. student reading proficiency. Second, FRE solely incorporates syllable, word, and sentence count and does not include possible other indicators of complexity. Statistically, FRE generates a point estimate but has no means to calculate standard errors to measure uncertainty. Lastly, the FRE scale does not carry inherent meaning and cannot be converted to a probability scale (Benoit et al., 2019).

Benoit et al. (2019) utilize the Bradley-Terry (BT) pairwise comparison approach to test and build a text complexity model that seeks to address these concerns. The BT model outputs the probability of victory for an entity in a pairwise competition (Benoit et al., 2019). The authors seek to leverage this functionality to develop a model that can: (1) when given two texts, evaluate the probability that one is easier; and (2) when given a text and a benchmark (such as a grade-level reading ability), evaluate the probability that the text is easier relative to the benchmark.

The authors constructed a corpus comprising 70 State of the Union (SOTU) speeches after 1950 (Benoit et al., 2019). They drew "snippets" of similar length from the corpus and hired coders to designate the easier text in each of 7,326 unique pairs of snippets. Because the authors required that each comparison be evaluated by multiple coders, the final sample included 19,810 pairwise comparisons.

In the main BT models the authors run, the results of the manual coding are used to construct the outcomes of interest (Benoit et al., 2019). The authors hypothesize that a complex political text will more frequently include long words, rare words, long sentences, and advanced structure. Accordingly, they construct model covariates to measure such elements, discussed in more detail in **Similarities & Differences**.

We replicated key findings from Benoit et al. (2019) and discuss our replication process in the following sections. First, we detail the analyses we replicated and compare our results with the authors' results.

Second, we discuss the functionality, organization, and reproducibility of the authors' code and note our adjustments. Last, we discuss potential extensions and substantive proposals for the paper.

II. Similarities & Differences

Benoit et al. (2019) currently host their data, documentation, RData files (.rda), and 13 scripts on Harvard Dataverse for users to reproduce their analyses (Benoit, 2019). In **Autopsy**, we discuss obstacles to running certain scripts and our reasons for running only a subset of the authors' analyses. For the current discussion of differences and similarities in results, it is important only to note that we focused on the code used to generate model covariates, perform covariate selection, run and evaluate key models, and visualize model results. Below, we discuss the comparison of our results to the authors' results for each of these components and provide details for how to cross-reference our results against the paper's results.

- *1. Covariate generation & covariate selection via random forest model*

We successfully followed the authors' code to compute potential model covariates. Based on their four assumptions of the composition of complex political text, the authors manually refined the list to 22 covariates across these four categories (Benoit et al., 2019). To identify the most predictive features of text ease, we replicated the authors' use of a random forest feature selection model (running both with and without bias reduction). We output the feature importance scores (based on node purity) and found that the most predictive feature in each of the four categories was (when running both with and without bias reduction):

- 1) Long words: `meanSentenceChars`
- 2) Rare words: `google_min_2000`
- 3) Long sentences: `W7C`
- 4) Advanced structure: `pr_noun`

These top features align with the top features presented in the "Bias Reduced" variable importance plot in **Appendix F** of the paper. However, the authors' "Not Bias Reduced" variable importance plot identifies `meanWordChars` as being slightly more important in the long sentences category than `W7C`. We do not replicate this result: in our non-bias reduced model, `W7C` still slightly edges out `meanWordChars` in our node purity results. The authors seem to opt for the non-biased reduced results and use `meanSentenceChars`, `google_min_2000`, `meanWordChars`, and `pr_noun` as the covariates in their "best" model, as displayed in the paper's **Table 2**. Thus, our results from feature selection only precisely match three of the four covariates they choose.

- *2. Models*

The authors build four main BT models (Benoit et al., 2019). The first model includes only FRE score as a covariate. The second used a re-weighted FRE in which the individual components of FRE score are included as covariates. The third uses three of the four covariates identified in feature selection (excepting `MeanWordChars`). The fourth uses all four selected covariates.

We successfully followed the code to replicate the four main models that the authors ran and our results yielded point estimates, standard errors, and AIC scores either identical or very close for all four models (compare ours in Step 6 here to **Table 2** in the paper). We likewise find that the fourth model performs best in terms of lowest AIC score.

- *3. Model performance visualization*

Finally, we replicated the visualization code to compare the results of the authors' model against the results using the original FRE score. Our replication (Step 7 here) produces the same figure corresponding to **Figure 1** in the authors' paper.

III. Autopsy

The replication package provided by the authors was comprehensive, well-organized, and well-documented. They added sequenced prefixes to their scripts (e.g., "01_", "02_", etc.) so that it was easy to identify the order in which to run the code. They published a `README.pdf` in which they clearly explained the organization of their replication materials, provided instructions on setting up R, detailed the purpose and component files of each script, and provided cautions about programs that took a particularly long time to run (Benoit, 2019). They also included a `codebook.pdf` in which they identified the variables included in each data file (Benoit, 2019).

While the authors produced all of their code in `.R` scripts, we opted to compile our replication code in a single R Quarto (`.qmd`) script. We decided to create a single script because our analyses altogether had a fairly short run time (< 10 minutes) and we felt it was easier to follow results when looking through a single file. We opted to prepare the replication in a `.qmd` file because we were able to knit to an HTML document which allowed for collation of code with results.

As mentioned in **Similarities & Differences**, there were several scripts or sections of scripts we did not run. Firstly, we did not run the first four scripts in the authors' sequence because these are used for preparing and uploading the data for crowdsourcing (Benoit, 2019). While it appears it would have been feasible to replicate the first three scripts (as all input data was available), the authors advise users to start with `03_generate_sentence_covariates.R` since users will not be able to replicate the intermediate fourth script of uploading and downloading the crowdsourced data (Benoit, 2019). We also opted not to run the sections of code that were flagged in `README.pdf` as having exceedingly long run times (e.g., bootstrapping) (Benoit, 2019). Additionally, we opted not to run any supplemental analyses (included in `11_supplemental_analysis.R`). Finally, the last section of the paper is devoted to applying the authors' developed model to various texts (Benoit et al., 2019). With the exception of producing Figure 1, we did not seek to replicate these results.

In general, the authors' neat and annotated code allowed us to smoothly replicate covariate generation, feature selection, model execution, and model visualization. The few errors we encountered were largely related to outdated packages. The authors wrote the `sophistication` package to execute many of their analyses, so it was imperative that we were able to install this library. The package is maintained on Benoit's GitHub repository (Benoit, n.d.). Potentially due in part to the fact that it has not been updated in 4 years, we initially had trouble installing the package; however, with several troubleshooting attempts we were ultimately able to install and use the package. Additionally, the authors used the `apsrtable` package to generate Table 2; however, upon encountering an error with this approach, we discovered that the package was no longer supported on the Comprehensive R Archive Network (CRAN) (CRAN, n.d.). This was a simple fix: we employed the `msummary()` function from the `modelsummary` package to produce an analogous table.

As noted above, some of the authors' code had significant run times. One such section of code in which the authors cautioned about run time was the execution of the unstructured BT models (Benoit, 2019). The results of these models are necessary to run the random forest models. Advantageously, the authors created a way to import the unstructured model results without having to re-run the model (which the authors estimated in `README.pdf` could take 5-6 hours): they included RData (`.rda`) files (e.g., `BT_unstructured_brT_abilities.rda` and `BT_unstructured_brF_abilities.rda`) which could be loaded into R (in under two minutes) and included the model results (Benoit, 2019).

We also made several elective updates in an effort to improve the code. While the authors tend to mostly use base R, we updated the code where possible to leverage more of the `tidyverse` syntax (e.g., chaining from `magrittr`, `select()` from `dplyr`). We also sought to make the code more robust by increasing the number of assertions (via `verify()` in `assertr`). We sought to increase the number of comments in our script and apply ample line breaks in the code to enhance readability. Lastly, in an effort to improve our own understanding of the data generation and manipulation process, we amplified the number of data displays in our output (e.g., **Section 2.1** in our analysis is a section we added to improve our understanding of the data).

IV. Extension

We noted several aspects of the authors' model validation process that we propose the authors could consider revisiting. First, we observed that the authors used the complete returned set of manually-coded data to train their model. We argue that this weakens their subsequent assertion that the model they develop is applicable to any political text. By omitting at least 20% of the manually-coded data and using it as testing data during model development, the authors would be in a position to make a stronger argument about how the model performs on unseen data in which the outcome label is known. Although the authors demonstrate in their paper how to apply the model to a political corpus (full SOTU speeches), this result is less compelling from a model validation standpoint because the "easiness" labels for the documents in this corpus are not known. Thus, while **Figure 2** in their paper presents an interesting result about the probability of the respective SOTU speech being easier than 5th-grade reading material, we find that it would be more credible with additional validation in the model training stage.

Similarly, we also noticed that the authors frame the results of their model using the FRE score as a benchmark (*"Experimenting with the continuous measure on the SOTU snippet corpus performs well in the sense that it returns point estimates on a roughly 0–100 scale commensurate (but not identical) to the FRE equivalents"* and *"within the (theoretical) minimum and maximum of the FRE range of 0–100, however, the correspondence is even higher. This implies that for the great majority of documents for which the FRE is used, our measure — preferred on theoretical grounds — is a good choice that will behave as expected"*) (Benoit et al., 2019). We found this positioning somewhat surprising given that the authors' motivation for the paper is in part the weaknesses of the FRE (Benoit et al., 2019). While it is certainly helpful to see how their model performs against the FRE, we felt that this comparison did not provide the validation argument implied in their paper.

Lastly, as mentioned above, toward the end of the paper the authors run their model on complete SOTU speeches (Benoit et al., 2019). Because the model was also trained on SOTU speeches, we believe it would be both informative and interesting to run the authors' model on a different type of political text, such as campaign speeches, city council meeting transcriptions, or White House press briefings. If additional funds allowed, we also propose that the authors use one of these different types of political texts to conduct another similar research project (including hiring manual coders) to see if the model results are similar to their model in this paper. Such a comparison, particularly if the results are closely aligned, would be another compelling model validation step.

References

Benoit, Kenneth, Kevin Munger, and Arthur Spirling. 2019. "Measuring and Explaining Political Sophistication through Textual Complexity." *American Journal of Political Science* 63 (2): 491–508.doi: 10.1111/ajps.12423

Benoit, Kenneth, 2019, "Replication Data for: Measuring and Explaining Political Sophistication Through Textual Complexity", https://doi.org/10.7910/DVN/9SF3TI, Harvard Dataverse, V1, UNF:6:3lWCX52gHXjVfaeDpmEBPQ==\[fileUNF\]

Benoit, Kenneth. (n.d.). Kbenoit/sophistication [R]. Retrieved February 17, 2025, from https://github.com/kbenoit/sophistication (Original work published 2017)

CRAN: Package apsrtable. (n.d.). Retrieved February 17, 2025, from https://cran.r-project.org/web/packages/apsrtable/index.html

GitHub repository

Our replication materials are available at https://github.com/mariabartlett/comp-ling-spring-2025-replication-1.