**JASA ACS Reproducibility Initiative - Author Contributions Checklist Form**

## Data

**Abstract (Mandatory)**

The data consist of two parts: (1) input data and (2) raw and pre-processing data. The input data are the data (i.e., frequency matrices) in RData format that can be directly used to reproduce the empirical results in the paper. The raw and pre-processing data contain the raw data (social media posts used in this study) and the resulting pre-processing data including the segmented words data and unique-word count data. The pre-processing data are generated from the raw data and are intermediate inputs used to produce the input data.

**Availability (Mandatory)**

The input (frequency matrices) data, together with the raw and pre-processing data (social media posts, labels, and all files generated by the pre-processing steps), are available as part of the supplementary material.

**Description (Mandatory if data available)**

In the supplementary material, we store all the datasets in four folders under the “Supplementary Data and Codes\data” directory.

The “Input” folder contains the following datasets in R data format.

1. “Alldata.RData” This is the frequency matrix representing the frequency counts of the words from the 12500 textual messages (Chinese social media posts) that we use for our statistical analysis in the paper. These 12500 posts are divided into five mutually exclusive subsets according to the location and time, labelled as Guangdong-Quarter 1, Guangdong-Quarter 2, Guangdong-Quarter 3, Guangdong-Quarter4, and Non-Guangdong, respectively.
2. “Count.RData” This is a list of unique words (a dictionary) that appear in the post data.
3. “dataGDQ1.RData,” “dataGDQ2.RData,” “dataGDQ3.RData,” “dataGDQ4.RData,” and “dataNGD.RData” These are the frequency matrices in sparse format representing the frequency counts of the words from the above-mentioned location-time datasets, Guangdong-Quarter 1, Guangdong-Quarter 2, Guangdong-Quarter 3, Guangdong-Quarter4, and Non-Guangdong, respectively.

These datasets in the “Input” folder are sufficient inputs to generate all the empirical results presented in the paper. For readers interested in knowing how to convert the raw Chinese textual data into the above data, we provide the raw and pre-processing data in two other folders, summarized as follows.

1. The “RawSamples” folder stores the raw posts before pre-processing in text format as “samples.txt” and the labels as “labels.txt”.
2. The “Preprocessing” folder contains five datasets in txt format produced by the pre-processing steps outlined in the paper and implemented by the associated code in the supplementary material. In particular, “result-samples.txt” is the result sentences after removing meta content from the raw data. “segment.txt” is the segmented words from the Stanford Segmenter. “postprocess-segment.txt” is the outcome after removing non-meaningful words from the segmented words. “count.txt” is a list of unique words. “matrix.txt” is the frequency matrix of words in each sample. The last two files can be directly imported into R to generate the input data needed for the empirical analysis in the paper.

Finally, for the convenience of reproducing results in this paper, we provide raw outputs beyond the summary statistics shown in the tables and figures of the paper. Specifically, we store the raw outputs of each empirical study (e.g., the results over many replications) in the “Numerical Outputs” folder.

## Code

**Abstract (Mandatory)**

Python scripts and bash shell scripts are provided for the pre-processing of the raw (social media posts) data into frequency matrices. R scripts are provided for all the analysis carried out in the paper based on the frequency matrices.

**Description (Mandatory)**

We provide two sets of code under the directory: “\Supplementary Data and Codes\code.” Directly under this directory are the R scripts that can be used to generate the empirical results in the paper. For the reproducibility purpose, we provide the code with fixed seeds. A detailed explanation of each individual R script can be found in the “Instruction for Use” section below.

In the “preprocess” folder, we provide the Python scripts and bash shell scripts, which generate the input data that are needed for the empirical analysis as well as the pre-processing data. These scripts are provided both as an .html file and an R Markdown file, allowing for readability by any user and for easy access and reproducibility. The code can be easily ported to another language such as R.

Code is licensed under the GPLv3 open source license and will be made public on GitHub. Code can be run on any modern PC.

**Version of Software and Packages**

Version of R: 3.3.3

Version of R packages:

nproc: 2.1.4

topicmodels: 0.2-8

e1071: 1.7-2

glmnet: 2.0-18

randomForest: 4.6-14

dsda: installed directly from the tar.gz file downloaded from website: “<https://ani.stat.fsu.edu/~mai/research.html>”

Version of Python: 3.5

Version of Python packages:

re: 2.2.1

nproc: 1.2

Version of Stanford Segmenter: 3.9.2

Available at: https://nlp.stanford.edu/software/segmenter.shtml

## Instructions for Use

**Reproducibility (Mandatory)**

The Python code and bash shell scripts (under “Supplementary Data and Codes\code\preprocess”) are provided as an R Markdown file, with a step-by-step guide. Using the R Markdown and the provided raw data will reproduce the frequency matrices and the list of unique words. The pre-processing steps read the raw data from samples.txt and then produce count.txt (a list of unique words) and matrix.txt (a frequency matrix).

The R codes, which are labelled by their purposes, are intended to produce all the tables and figures in the paper, except for Figure 3 which is adapted from a published work and Figure 4 which is a flow chart. Below is a description of each individual R code in the file.

ImportData.R - Import the list of unique words, the frequency matrices from the pre-processing steps, and the labels of the raw data into R, then save the results as five frequency matrices (dataGDQ1.RData, dataGDQ2.RData, dataGDQ3.RData, dataGDQ4.RData, and dataNGD.RData) as well as the list of unique words (count.RData). These data files are also provided in the "data\Input" folder for the convenience of readers who do not want to run this code.

SearchingK.R - Produce the topics and keywords for each topic.

NPsLDAeffect.R - Produce the topics and the corresponding coefficients generated by NP-sLDA.

umbrellaAlg.R - The NP-umbrella algorithm illustrated in Figure 3.

Figure1.R, Figure2.R, and Table1.R-Table8.R - Generate each individual figure or table in the paper.

**Computational Speed (for each empirical result and each computation step)**

Below, we provide the approximate running time for generating each empirical result, specifically, Tables 2-8, per replication. We also reported detailed time per replication for the Latent Dirichlet Allocation (LDA) step and the NP Umbrella Algorithm step (includes running all methods NP-sLDA, NP-RF, NP-SVM, NP-NB, NP-PLR). Please note that the results of Tables 2, 3, 4, and 7 presented in the paper are based on 100 replications.

Run on iMac 2019, Processor 3.6GHz Intel Core i9, Memory 64GB 2667 MHz DDR4

Table 2:

average total running time (including LDA, Umbrella Algorithm, data shuffling and constructing training and testing data) ~ 5.81 min

average running time for LDA ~ 2.47s

average running time for the Umbrella Algorithm ~ 5.73 min

Table 3:

average total running time (including LDA, Umbrella Algorithm, data shuffling and constructing training and testing data) ~ 5.23 min

average running time for LDA ~ 2.31s

average running time for the Umbrella Algorithm ~ 5.15 min

Table 4:

average total running time (including LDA, Umbrella Algorithm, data shuffling and constructing training and testing data) ~ 5.87 min

average running time for LDA ~ 2.54s

average running time for the Umbrella Algorithm ~ 5.79 min

Table 5:

total running time (including LDA, NP-sLDA, constructing training and testing data based on LDA results) ~ 54.79s

running time for LDA ~ 4.85s

running time for NP-sLDA ~ 30.54s

Table 6:

total running time (including LDA, NP-sLDA, constructing training and testing data based on LDA results) ~ 54.62s

running time for LDA ~ 4.92s

running time for NP-sLDA ~ 30.11s

Table 7:

average total running time (including LDA, Umbrella Algorithm, data shuffling and constructing training and testing data) ~ 26.60 min

average running time for LDA ~ 10.31s

average running time for the Umbrella Algorithm ~ 26.28 min

Table 8:

total running time (including LDA, NP-sLDA, constructing training and testing data based on LDA results) ~ 1.35min

running time for LDA ~ 21.13s

running time for NP-sLDA ~ 53.50s

## Notes

Reviewers may access codes and data from the attached zip files submitted along with the paper.