Tainted Trends

Pre-Registration

Hauke Roggenkamp Christian Hildebrand

November 10, 2023

Table of contents

---

1 Motivation

In today's hyperconnected world, brands are co-created by a variety of stakeholders (Swaminathan et al. 2020). We study exogenously inflicted brand dilution caused by stakeholders who *hijack* brand experiences in social media (Grier et al. 2010). Specifically, we investigate the contaminating effect (Whitler, Besharat, and Kashmiri 2021) of stakeholders who append their (from a brand's perspective) unfavorable content to a branded hashtag.

2 Research Question

Does the exposure to hijacked content in the context of branded hashtags dilute trust in the associated brand?

3 Hypotheses

The more hijacked content a participant is exposed to, the lower her trust in the brand. (See Section 9 for the exact operationalization.)

4 Design

We will expose participants to a realistic social media feed that covers the *National Fried Chicken Day*. We will ask participants to scroll through the feed to learn more about that event and tell them that it is associated to the brand *Kentucky Fried Chicken (KFC)*. Importantly, we will exogeneously manipulate the degree to which a feed is hijacked by randomly[1] drawing a set of tweets that are favorable and unfavorable (from a brand's perspective). An unfavorable tweet may consist of animal rights, labor rights or pro-vegan content that is appended to the hashtag *#NationalFriedChickenDay*. Favorable content also uses that hashtag but appends

[1]We draw tweets randomly on a participant level without replacement. This not only implies a random composition of feeds but also a random order of tweets within that feed.

content showing recipes, family gatherings or actual meals. You can find the complete list of tweets we sample from here on OSF or in the archived `*.otreezip` (more on that below).

Subsequently, we will ask participants to describe their experience, thoughts and feelings towards the focal brand (KFC) in an open text field[2] and elicit an established brand trust measure using four seven-point semantic-differentials to measure how much a participant believes the brand is honest and not manipulative (Kirmani et al. 2017)[3]. We will also ask participants whether they like KFC on a single seven-point Likert scale. Furthermore, we will elicit covariates covering the participants' fast food and meat consumption as well as their age, gender and education.

You can find the exact materials on OSF. This study's software is archived in oTree's (Chen, Schonger, and Wickens 2016) native zip format here.

5 Implementation

We designed an oTree experiment that mimics social media feeds. In this study, we will mimic a twitter feed where participants can scroll through and interact with the feed as well as individual tweets. You can find more information about the software here.

6 Population and Sample Size

We will recruit participants from Prolific who meet the following criteria:

- Approval Rate >= 99%
- First Language == 'English'
- Location == ('USA' | 'Canada')

Because the software is computationally expensive, we recruit 300 participants (and not more) to ensure a smooth experience for the participants.

7 Preprocessing

This section describes how we preprocess the data (using R and quarto) in detail. You can skip this section and jump to the exclusion criteria in Section 8 or the analysis Section 9.

[2] *"Please describe as detailed as you can: What were the first thoughts and feelings towards the brand KFC that came to your mind when you were scrolling through the feed?"*

[3] dishonest / honest, insincere / sincere, manipulative / not manipulative, not trustworthy / trustworthy.

7.1 Install Packages

We install the following packages using the `groundhog` package manager to increase computational reproducibility.

```
if (!requireNamespace("groundhog", quietly = TRUE)) {
    install.packages("groundhog")
    library("groundhog")
}

pkgs <- c("magrittr", "data.table", "knitr", "stringr", "jsonlite",
          "ggplot2", "patchwork",
          "stargazer",
          "tidytext", "textdata")

groundhog::groundhog.library(pkg = pkgs,
                             date = "2023-09-25")

rm(pkgs)
```

7.2 Read Data

For this preregistration, we read some data we generated while testing the software. The eventual analysis will read a different data set that we will store in `../data/raw/` once we have collected the data.

```
raw <- data.table::fread(file = "../data/testing/all_apps_wide-2023-11-09.csv")
```

7.3 Select Data

```
cols <- str_detect(string = names(raw),
                   pattern = "participant.label|participant.code|liked_item|reply_to_item|

data <- raw[, ..cols]
```

7.4 Rename Data

```
names(data) %>%
  str_replace_all(pattern = ".*player\\.", replacement = "") %>%
  str_replace_all(pattern = "\\.", replacement = "_") %>%
  str_to_lower() %>%
  setnames(x = data)
```

7.5 Number or Fraction of Unfavorable Tweets

```
feed_composition <- data[,
                         .(favorable_count = favorable_feed %>%
                             strsplit(split = ',') %>%
                             unlist() %>%
                             as.numeric() %>%
                             sum(na.rm = TRUE),
                           total_count = favorable_feed %>%
                             strsplit(split = ',') %>%
                             unlist() %>%
                             length()),
                         by = participant_code]

feed_composition[, fraction_unfavorable := ((1 - favorable_count / total_count)*100)]
```

7.6 Treatment Info by Tweet

```
condition <- data[,
                  .(favorable = favorable_feed %>%
                      strsplit(split = ',') %>%
                      unlist() %>%
                      str_replace(pattern = ' ',
                                  replacement = '') %>%
                      as.character() %>%
                      as.numeric() %>%
                      as.logical()),
                  by = participant_code]

condition[, displayed_sequence := 1:.N, by = participant_code]
```

7.7 Item Sequence

```
sequence_list <- lapply(seq(nrow(data)), function(i) {
  participant_code <- data[i, participant_code]
  sequence <- data[i, sequence]
  tmp <- data.table(participant_code = rep(participant_code,
                                           length(unlist(strsplit(sequence, ", ")))),
                    tweet = unlist(base::strsplit(x = sequence,
                                                  split = ", ")) %>%
                      as.integer())
  tmp[, displayed_sequence := 1:.N]
  return(tmp)
})

# Combine the list of data.tables into one data.table
display <- rbindlist(sequence_list)

rm(list = c('sequence_list', 'tmp'))
```

7.8 Scroll Sequence

Because people may scroll back and forth, this sequence may be longer than the number of tweets, simply because some tweets may occur several times in that scroll sequence.

```
scroll <- data[,
               .(participant_code,
                 tweet = scroll_sequence %>%
                   strsplit(split = '-') %>%
                   unlist() %>%
                   str_replace_all(pattern = 'i',
                                   replacement = '')),
               by = participant_code][!(tweet == shift(tweet, type = "lead")),
                                      by = participant_code][,
                                                             scroll_sequence := 1:.N,
                                                             by = participant_code]



data[1, .(participant_code,
        item = scroll_sequence %>%
```

```
          strsplit(split = '-') %>%
          unlist() %>%
          str_replace_all(pattern = 'i', replacement = ''))][!(item == shift(item, type =
```

7.9 Viewport

```
viewport <- data[,
                 viewport_data %>%
                     str_replace_all(pattern = '"""',
                                     replacement = '"') %>%
                     fromJSON,
                 by = participant_code][!is.na(doc_id)]

# sum of durations by tweet (in case someone scrolled back and forth)
viewport <- viewport[,
                     .(seconds_in_viewport = sum(duration,
                                                 na.rm = TRUE)),
                     by = c('participant_code', 'doc_id')]


# rename
setnames(x = viewport,
         old = 'doc_id',
         new = 'tweet')
```

7.10 Reactions

```
# make sure each and every reply column is a string
sdcols <- names(data) %>%
  str_subset(pattern = "reply_to_item")
data[,
     paste0(sdcols) := lapply(.SD, as.character),
     .SDcols = sdcols]

# melt data
reactions <- data.table::melt(data = data,
                              id.vars = c('participant_code', 'participant_label'),
                              measure.vars = patterns('liked_item_', 'reply_to_item'),
```

```
                                variable.name = 'tweet',
                                value.name = c('likes', 'replies'))

# make tweet identifier numeric for compatibility reasons.
reactions[, tweet := as.numeric(as.character(tweet))]
```

7.11 Self-Reports

```
# select
cols <- str_detect(string = names(raw),
                   pattern = "participant\\.code|trust|usability|KFC|meatless|\\.age|gende

self_reports <- raw[, ..cols]

# rename
names(self_reports) %>%
  str_replace_all(pattern = ".*player\\.", replacement = "") %>%
  str_replace_all(pattern = "\\.", replacement = "_") %>%
  str_to_lower() %>%
  setnames(x = self_reports)
```

```
# calculate simple indices

self_reports[, trust := (trust_1 + trust_2 + trust_3 + trust_4)/4]
self_reports[, usability := (usability_1 + usability_2 + usability_3)/3]

# transform gender variable
self_reports[, female := 1]
self_reports[gender != 1, female := 0]

# ... and meatless
self_reports[, vegetarian := 0]
self_reports[meatless == "Yes", vegetarian := 1]
```

7.12 Merge

Note that the last item displayed never leaves the viewport which is why we do not measure the duration it is displayed.

```r
# merge 1: displayed tweets with condition info
long <- display[condition, on = .(participant_code, displayed_sequence)]

# merge 2: long with viewport data (full join)
long <- rbind(
  fill = TRUE,
  long[viewport, on = .(participant_code, tweet)],  # join
  long[!viewport, on = .(participant_code, tweet)]  # anti-join
)

# merge 2: long with reactions (left join)
long <- long[reactions, on = .(participant_code, tweet)]

# merge 3: long with reactions (left join)
long <- long[feed_composition, on = .(participant_code)]

# merge 4: long with self-reports
long <- long[self_reports, on = .(participant_code)]


# remove items that haven't been displayed
tweets <- long[!is.na(displayed_sequence)]

# order
setorder(x = tweets, participant_code, displayed_sequence)
```

7.13 Individual Level Data

```r
individuals <- tweets[, -(2:8)] %>% unique()
```

7.14 Write Data

```r
fwrite(x = tweets,
       file = '../data/processed/tweet_level_data.csv')
fwrite(x = individuals,
       file = '../data/processed/tweet_level_data.csv')
```

7.15 Clean up

Lastly, we clean the global environment.

```
objects <- c(ls(), "objects", "objects_to_keep", "objects_to_remove")
objects_to_keep <- c("raw", "tweets", "individuals")
objects_to_remove <- setdiff(objects, objects_to_keep)
rm(list = objects_to_remove)
```

8 Exclusion

We exclude participants who self-reported that they used a generative AI to answer our questions. In addition, we only focus on participants who completed the survey.

```
individuals <- individuals[completed_survey == TRUE & genai == "No"]
tweets <- tweets[participant_code %in% individuals[, participant_code %>% unique()]]
```

9 Primary Analyses

9.1 Trust

In this study, we focus primarily on trust. We analyze the data in three steps: First, we summarize the data descriptively.

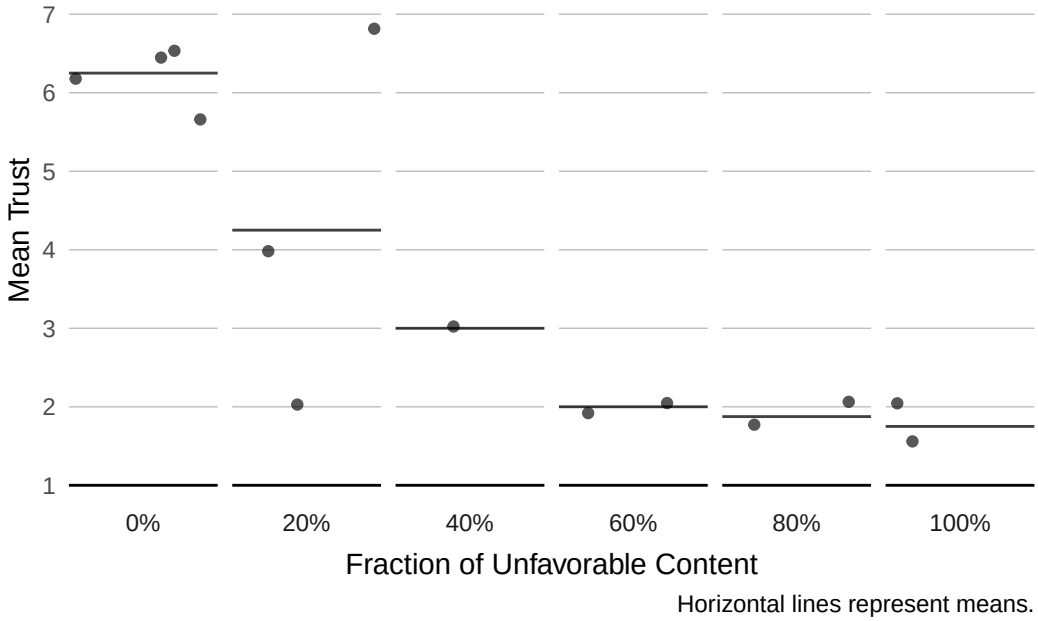
Table 1: Summary Statistics for Trust Measure by Feed Composition

fraction_unfavorable	mean_trust	sd_trust	count
0	6.250	0.3535534	4
20	4.250	2.3848480	3
40	3.000	NA	1
60	2.000	0.0000000	2
80	1.875	0.1767767	2
100	1.750	0.3535534	2

Second, we visualize the data.

Third, we run OLS regressions in three models. The first model tests the main effect, the second controls for the participant's diet and whether she enjoys KFC's food. The third

Figure 1: Level of Trust by Feed Composition



model adds interaction terms as the effect of hijacked content may differ between vegetarians and non-vegetarians, for instance.

We will also add another model that is not displayed here, where we control for the remaining covariates.

% Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac at gmail.com % Date and time: Fri, Nov 10, 2023 - 14:54:41

Model 4, 5, and 6 are very similar to the first three models but contain a different dependent variable: `kfc_brand` — the degree to which a participant likes the brand KFC.

9.2 Sentiment Analyses

Our test data does not contain any real text data, which is why we cannot operationalize the exact analysis we will run. Hence, the following code provides the most basic basic sentiment analysis we can imagine. It is based on unigrams, the `AFINN` general-purpose lexicon (Nielsen 2011) and without stemming.

```
tidy_text <- unnest_tokens(tbl = individuals,
                           output = word,
                           input  = perception,
                           token = 'words')
```

Table 2: Regression Table - Level of Trust and Brand Liking by Feed Composition

	Dependent varia			
	trust			
	(1)	(2)	(3)	
fraction_unfavorable	−0.047***	−0.045***	0.009	
	(0.009)	(0.010)	(0.044)	
vegetarian		1.598	3.592	
		(1.586)	(2.513)	
kfc_food		0.217	0.606	
		(0.351)	(0.472)	
fraction_unfavorable:vegetarian			−0.048	
			(0.040)	
fraction_unfavorable:kfc_food			−0.011	
			(0.009)	
Constant	5.677***	4.439**	2.302	
	(0.484)	(1.931)	(2.627)	
Observations	14	14	14	
R^2	0.710	0.739	0.783	
Adjusted R^2	0.686	0.660	0.647	
Residual Std. Error	1.198 (df = 12)	1.245 (df = 10)	1.269 (df = 8)	0
F Statistic	29.345*** (df = 1; 12)	9.427*** (df = 3; 10)	5.767** (df = 5; 8)	14.0

Note:

```
data(stop_words)
stop_words <- stop_words %>% data.table()
custom_stop_words <- c('tweet', 'tweets', 'post', 'posts', 'feed', 'KFC')
stop_words <- rbind(stop_words,
                    data.table(word = custom_stop_words,
                               lexicon = 'custom'))
```

```
tidy_text <- tidy_text[!stop_words, on = .(word)]
```

```
sentiments <- get_sentiments("afinn") %>% as.data.table()
afinn <- tidy_text[sentiments, on = .(word)]

sent <- afinn[,
              .(sentiment = sum(value)),
              by = c('participant_code', 'fraction_unfavorable')]

# merge
sentiment <- individuals[sent, on = .(participant_code)][!is.na(participant_code)]

# clean up
rm(list = c("sent", "sentiments", "tidy_text", "stop_words"))
```

We envision the same visualization and regression as in the previous sub-section:

% Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac at gmail.com % Date and time: Fri, Nov 10, 2023 - 14:54:42

In addition to that, we plan to use the same data for topic modeling.

Chen, Daniel L., Martin Schonger, and Chris Wickens. 2016. "oTree-an Open-Source Platform for Laboratory, Online, and Field Experiments." *Journal of Behavioral and Experimental Finance* 9: 88–97. https://doi.org/10.1016/j.jbef.2015.12.001.

Grier, Chris, Kurt Thomas, Vern Paxson, and Michael Zhang. 2010. "@Spam: The Underground on 140 Characters or Less." In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, 27–37. CCS '10. New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/1866307.1866311.

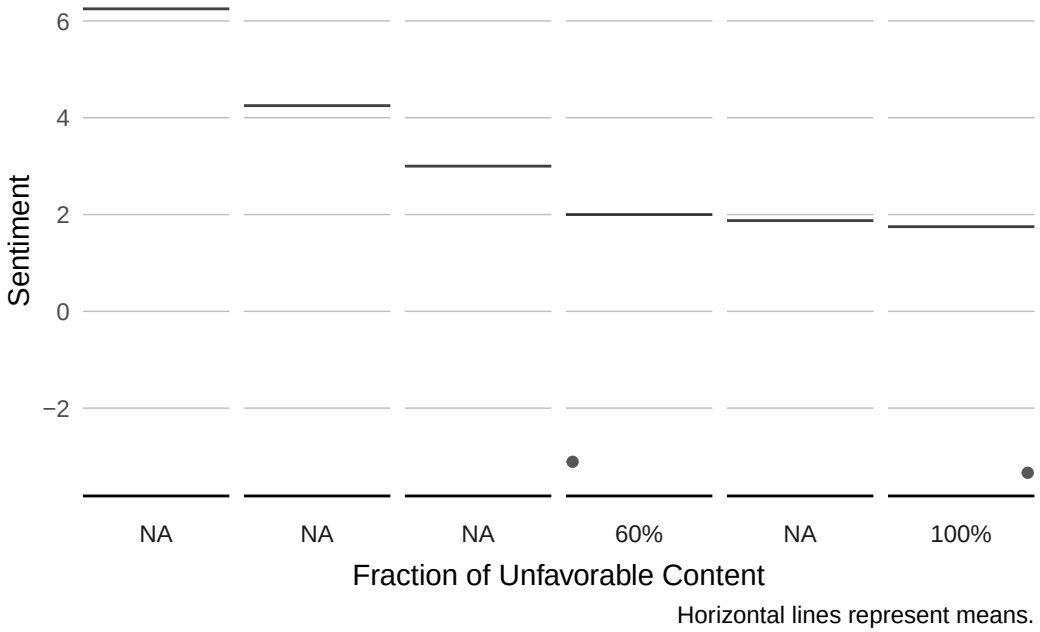
Kirmani, Amna, Rebecca W. Hamilton, Debora V. Thompson, and Shannon Lantzy. 2017. "Doing Well Versus Doing Good: The Differential Effect of Underdog Positioning on Moral and Competent Service Providers." *Journal of Marketing* 81 (1): 103–17. http://www.jstor.org/stable/44879037.

Nielsen, Finn Årup. 2011. "A New ANEW: Evaluation of a Word List for Sentiment Analysis in Microblogs." https://arxiv.org/abs/1103.2903.

Table 3: Regression Table - Sentiment by Feed Composition

	Dependent variable:		
	sentiment		
	(1)	(2)	(3)
fraction_unfavorable	0.000	0.000	0.000
vegetarian			
kfc_food			
fraction_unfavorable:vegetarian			
fraction_unfavorable:kfc_food			
Constant	−3.000	−3.000	−3.000
Observations	2	2	2
Note:	*p<0.1; **p<0.05; ***p<0.01		

Figure 2: Sentiment by Feed Composition



Swaminathan, Vanitha, Alina Sorescu, Jan-Benedict E. M. Steenkamp, Thomas Clayton Gibson O'Guinn, and Bernd Schmitt. 2020. "Branding in a Hyperconnected World: Refocusing Theories and Rethinking Boundaries." *Journal of Marketing* 84 (2): 24–46. https://doi.org/10.1177/0022242919899905.

Whitler, Kimberly A., Ali Besharat, and Saim Kashmiri. 2021. "Exogenous Brand Crises: Brand Infection and Contamination." *Marketing Letters* 32 (1): 129–33. https://doi.org/10.1007/s11002-020-09554-4.