Social Engineering and Text Generation without Neural Networks

Stefan Stein

University of Warwick

10/30/2019

What is this talk about?

- Based on 2016 DEF CON Talk "Weaponizing Data Science for Social Engineering" by John Seymour, Philip Tully (Seymour and Tully 2016)
- Credit for method and some of the images goes to the original authors!
- They introduced the *SNAP_R* tool: Social Network Automated Phishing with Reconnaissance
- Use NLP for automated spear phishing people of interest on Twitter

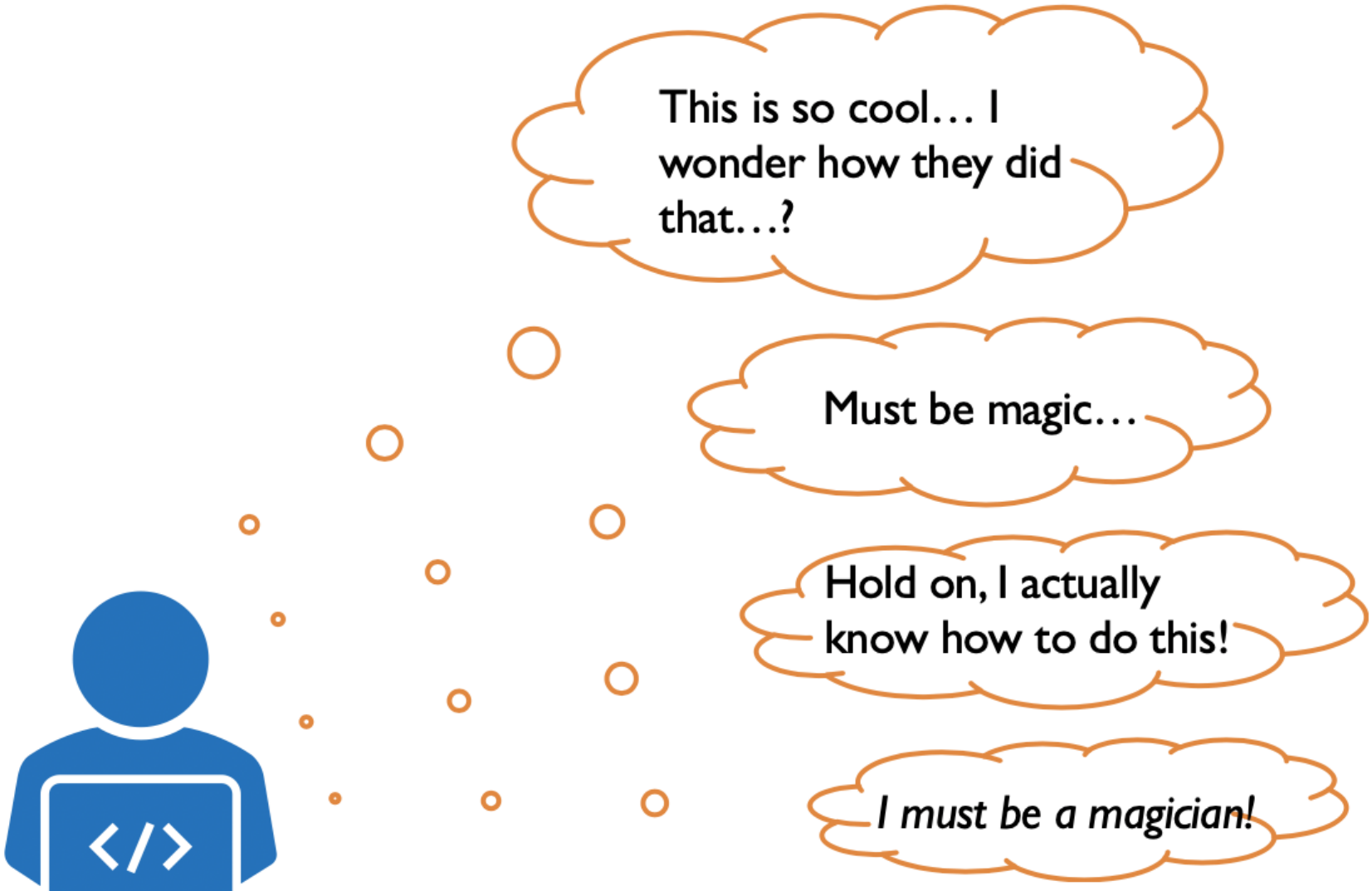
Who am I and why am I giving this talk?

- PhD student in statistics at the University of Warwick
- High-dimensional statistics and networks, but also very interested in security
- This is joint work with David Selby (@TeaStats)

WARWICK
THE UNIVERSITY OF WARWICK

Me watching something on cybersecurity

But sometimes...



The phishing game

You click on the link, you lose.

Types of phishing

- Phishing: Mostly automated, 5-14% success rate. *Low effort, low success rate*
- Spear phishing: highly manual, 45% success rate. *High effort, high success rate*
- SNAP_R: automated spear phishing, >30% success rate.

(percentages taken from original presentation)

Why Twitter?

- Bot friendly API: 1 line of code gets me
  - The last couple of 1000 tweets of a user or
  - All their followers or
  - All the people following them etc.
- Users not suspicious of shortened urls due to character limit
- "Twitter-speak" is not English. We can get away with bad grammar.
- We all know not to click on suspicious email links... on social media people are much more trusting
- Culture of sharing things... which can be used against you

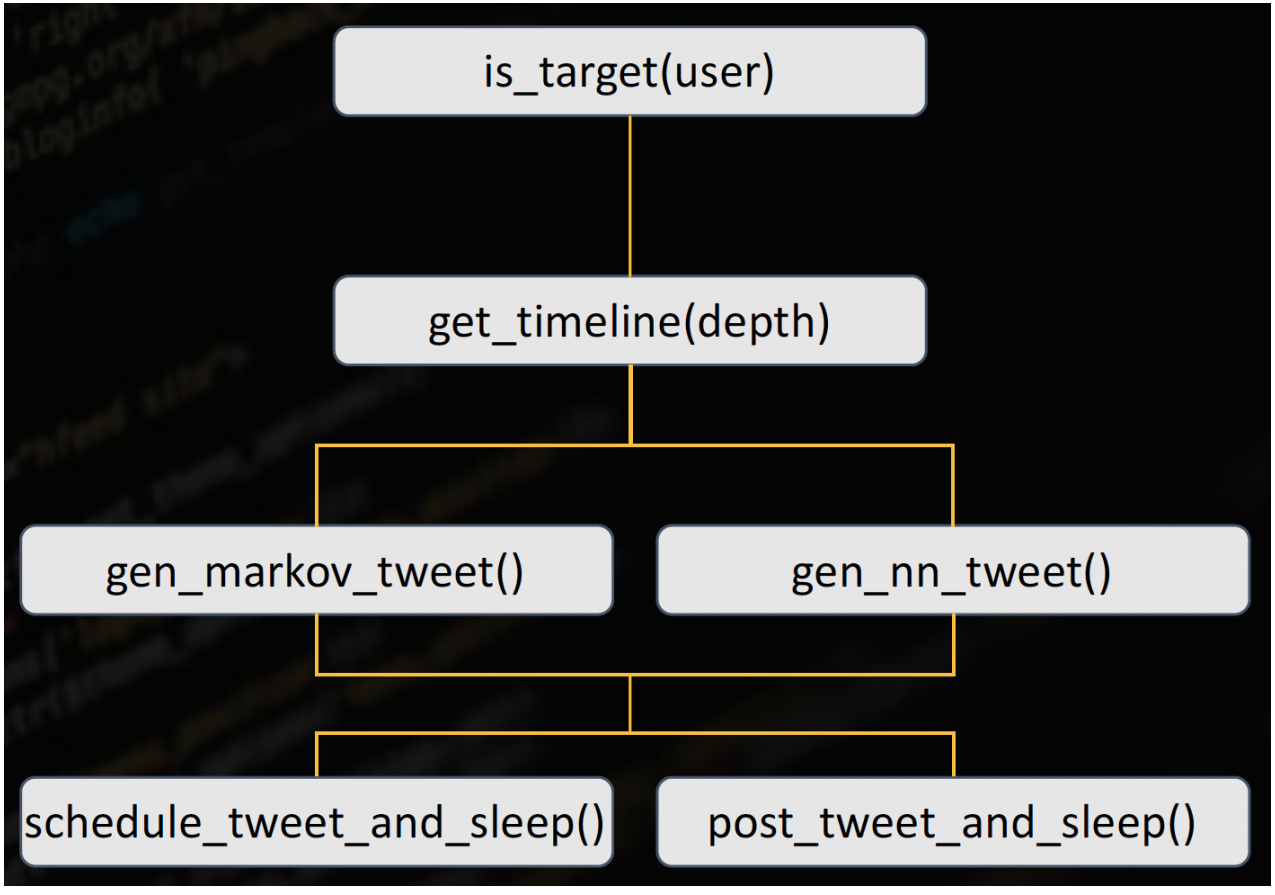
Phases of SNAP_R

- Phase 1: Automated target discovery
  - Input: List of users
  - Output: High-value targets we want to spear phish
- Phase 2: The phishing
  - Craft message with malicious link for each target

Phase 1: Automated target discovery

- Description field: job title, interests
- Number of followers/ followings
- Have they changed the default settings?
- Use clustering algorithms to find similar high-value targets

Phase 2: Automated Social Spear Phishing



(image taken from the original presentation)

Twitter API gives us times of activity of user. Schedule the tweet in most active hour.

WARWICK
THE UNIVERSITY OF WARWICK

Phase 2: Markov Chain vs Recurrent Neural Network

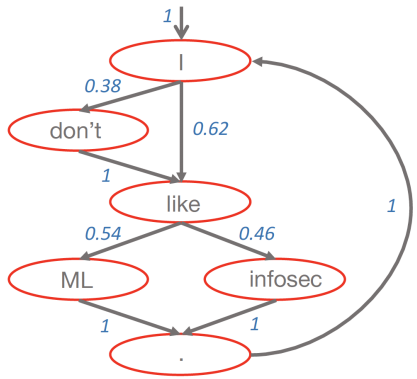
Metric \ Model	LSTM	Markov Chain
Training Speed	Days	Seconds
Accuracy	High	Medium
Availability	Public	Public
Size	Large	Small
Caveats	• Deeper representation of natural language, generalizes well • Retraining required for new languages	• Overfits to each user, can create temporally irrelevant tweets • Performs poorly on users with few tweets

(image taken from the original presentation)

WARWICK
THE UNIVERSITY OF WARWICK

A closer look: Markov Chains

- Markov Chain: Given current word, what is the most likely next word?



(image taken from original presentation)

- They only use bi-grams and still get a click-through rate above 30%!

Enter the statistician

- R packages `rtweet` for querying Twitter API, `httr` for web requests, `jsonlite` for handling JavaScript Object Notation (JSON).
- Use @realDonaldTrump as example user.
- We can use the Trump Twitter Archive

Enter the statistician

Load Trump tweets from the Trump Twitter Archive and load it into `archive_df`. For other users: Call to Twitter API is a one-liner and gives latest 3,200 tweets of user.

```
# Adapted from https://github.com/mkearney/trumptweets
get_trump_archive <- function(year) {
  url <- paste0('http://trumptwitterarchive.com/',
                'data/realdonaldtrump/',
                year,
                '.json')
  response <- httr::GET(url, httr::accept_json())
  httr::warn_for_status(response)
  text <- httr::content(response, 'text', encoding = 'utf8')
  jsonlite::fromJSON(text)
}
archive_list <- lapply(2009:2019, get_trump_archive)
archive_df <- do.call(rbind, archive_list)
```

What did we get?

Some Trump tweets:

created_at	text
2019-05-12 10:34:06	Under the leadership of @Potus Trump our economy is roaring. Businesses large and small have created more than 5.8 million jobs, unemployment is at a nearly 50-year low, and there are more Americans working today than ever before in the history of this country! https://t.co/417MoB5FVm
2019-05-12 10:33:41	“The Democrats have nothing. Just want to distract from this President. The FBI was not doing its job, the State Department was covering things up everyday for Hillary. At the end of the day they’re fearful of what they did, and should be fearful. This is a tough.....

Markov Chain based on Bi-grams

```
library(dplyr)
library(tidytext)

MC_1step <-
  archive_df %>%
  distinct(status_id, .keep_all = TRUE) %>%
  transmute(status_id,
            text = gsub("[^[:alnum:][:space:]@#]", "", text),
            text = gsub("http[[:alnum:][:punct:]]*", "", text),
            text = paste('THISISTHESTART', text, 'THISISTHEENDMYFRIEND')) %>%
  unnest_tokens(tweet, text, token = 'tweets',
                to_lower = FALSE,
                strip_punct = FALSE,
                strip_url = TRUE) %>%
  group_by(status_id) %>%
  do(data.frame(word1 = head(.$tweet, -1),
                word2 = tail(.$tweet, -1),
                stringsAsFactors = FALSE)) %>%
  ungroup() %>%
  count(word1, word2) %>%
  filter(!(word1 == 'THISISTHESTART' & word2 == 'THISISTHEENDMYFRIEND')) %>%
  group_by(word1) %>%
  mutate(prop = n / sum(n)) %>%
  ungroup()
```

Markov Chain based on Bi-grams

This is what `MC_1step` looks like:

word1	word2	n	prop
The	ceremony	1	0.0016611
now	#	1	0.0039370
Jeff	Flakey	1	0.0357143
of	West	3	0.0013812
and	Ron	1	0.0003902
pretty	IG	1	0.3333333

Sample from the Markov Chain

Sample from MC until we reach `max_length` or the end-word `THISISTHEENDMYFRIEND`.

```
simulate1 <- function(max_length, dataset = MC_1step) {
  chain <- vector(mode = 'character', length = max_length)
  chain[1] <- 'THISISTHESTART'
  for (i in 2:max_length) {
    pairs <- filter(dataset, word1 == chain[i-1])
    if (nrow(pairs) == 0)
      break
    chain[i] <- sample(pairs$word2, 1, prob = pairs$prop)
    if (stringr::str_detect(chain[i], 'THISISTHEENDMYFRIEND')) {
      chain[i] <- ''
      break
    }
  }
  return(gsub(' .*$', '', chain[-1]))
}
simulate1(20)
```

```
##  [1] "The"       "Democrats" "and"       "Drugs"     "and"
##  [6] "dirtiest"  "political" "trick"     "in"        "our"
## [11] "Southern"  "Border"    "must"      "have"      "all"
## [16] "the"       "citizens"  "deserve"   ""
```

Sample from the Markov Chain

```
generate_tweets <- function(n = 1, max_length = 20, dataset = MC_1step) {
  replicate(n, paste(simulate1(max_length, dataset), collapse = ' '))
}
```

Generating tweets like this gives:

1	All part of my approval rating among other President Putin of great relations with North Korea Kim I support
2	Melania Trump Campaign began 19 with Turkey We are negotiating in Florida I have She is a National Emergency
3	Congratulations on them for the meeting this very liberal democrats

Can we do better?

- Not satisfactory yet: Grammar is all over the place; not coherent.
- Try a two-step Markov Chain using tri-grams: Look at two previous words
- Sample space is *much* larger now and code is slower. But still takes less than 1 min, i.e. still much faster than LSTMs.

Two-step Markov Chain: Code 1

```r
# Helper function
twostep_transitions <- function(vec) {
  onestep <- cbind(head(vec, 1), tail(vec, -1))
  pairs <- apply(onestep, 1, paste, collapse = ' ')
  data.frame(word1 = head(pairs, -1),
             word2 = tail(pairs, -1),
             stringsAsFactors = FALSE)
}
```

Two-step Markov Chain: Code 2

```
MC_2step <-
  archive_df %>%
  #sample_n(2000) %>%
  distinct(status_id, .keep_all = TRUE) %>%
  transmute(status_id,
            text = gsub("[^[:alnum:][:space:]@#]", "", text),
            text = gsub("http[[:alnum:][:punct:]]*", "", text),
            text = paste('THISISTHESTART', text, 'THISISTHEENDMYFRIEND')) %>%
  unnest_tokens(tweet, text, token = 'tweets',
                to_lower = FALSE,
                strip_punct = FALSE,
                strip_url = TRUE) %>%
  group_by(status_id) %>%
  do({
    onestep <- cbind(head(.$tweet, -1), tail(.$tweet, -1))
    tmp <- apply(onestep, 1, paste, collapse = ' ')
    data.frame(word1 = head(tmp, -1),
               word2 = tail(tmp, -1),
               stringsAsFactors = FALSE)
  }) %>%
  ungroup() %>%
  count(word1, word2) %>%
  group_by(word1) %>%
  mutate(prop = n / sum(n)) %>%
  ungroup()
```

Two-step Markov Chain: Transition probabilities

word1	word2	n	prop
results help	help Trump	2	1.0
fair deal	deal is	1	0.5
occasion in	in which	1	1.0
ISIS until	until I	1	1.0
Fighters FEMA	FEMA and	1	1.0

Two-step Markov Chain: Sampling

Sampling from the two-step Markov Chain:

1	Senator Bob Casey Lou is tough on crime and our Country can achieve amazing breakthroughs when we put politics aside
2	extraordinary growth economic success and riches under the pressure Dont let Biden take us backwards
3	Angel mom demands Trumps wall

Smoothing

- Looks ok so far, but *overfit* to Donald Trump's tweets, i.e. we can only generate a mixture of what he already said.
- To make the MC use new words: Pull tweets from verified accounts (here: around 3,200 tweets from NY Times) and mix with tweets from target.

The smoothing procedure

Intuitively:

- Create two MCs, one for the target, one for the verified tweets: `MC_target`, `MC_verified`.
- Pick a smoothing parameter `alpha` and sample from the convex combination of the two MCs: `alpha*MC_target + (1-alpha)*MC_verified`. That is
  - Multiply transition probabilities of `MC_target` by `alpha` and those of `MC_verified` by `1-alpha`. Sample from the resulting MC.
  - Intuitively, at each time point, given the current word `w_k`, we sample the next word from `MC_target | w_k` with probability `alpha` and from `MC_verified | w_k` with probability `1-alpha`.

Smoothing: Code 1

These are 3199 tweets from NY Times.

```
verified_tweets <- readr::read_csv('verified_tweets.csv')
```

```
MC_verified2 <- verified_tweets %>%
  distinct(status_id, .keep_all = TRUE) %>%
  transmute(status_id,
            text = gsub("[^[:alnum:][:space:]@#]", "", text),
            text = gsub("http[[:alnum:][:punct:]]*", "", text),
            text = paste('THISISTHESTART', text, 'THISISTHEENDMYFRIEND')) %>%
  unnest_tokens(tweet, text, token = 'tweets',
                to_lower = FALSE,
                strip_punct = FALSE,
                strip_url = TRUE) %>%
  group_by(status_id) %>%
  do({
    onestep <- cbind(head(.$tweet, -1), tail(.$tweet, -1))
    tmp <- apply(onestep, 1, paste, collapse = ' ')
    data.frame(word1 = head(tmp, -1),
               word2 = tail(tmp, -1),
               stringsAsFactors = FALSE)
  }) %>%
  ungroup() %>%
  count(word1, word2) %>%
  group_by(word1) %>%
  mutate(prop = n / sum(n)) %>%
  ungroup()
```

Smoothing: Code 2

```
smooth2 <- function(alpha) {
  trump_transitions <- transform(MC_2step, prop <- prop * alpha)
  other_transitions <- transform(MC_verified2, prop <- prop * (1 - alpha)) #%>%
  rbind(trump_transitions, other_transitions) %>%
    group_by(word1, word2) %>%
    summarise(prop = sum(prop)) %>%
    ungroup()
}
```

Smoothing: 50% Trump, 50% NYT

```
generate_tweets2(n = 5, max_length = 20, dataset = smooth2(0.5))
```

1	Good Morning Have A Great Day
2	Again and again and again and big investments in podcasting helped narrow the companys CEO for being a Russian motel
3	Case closed McConnells speech was quickly criticized by top Democrats
4	Statement on the expected impact of Hurricane #Florence make sure were modernizing and keeping up with Comeys accounts according to
5	Except the results will determine how long they take Dont quit before the anniversary of the report From @SecretsBedard Kushner

Potential Use cases

- Social Media security awareness and education
- Automated internal testing of employee awareness
- Social engagement
- Recruiting?

Conclusions

- AI/ ML will be used offensively in cybersecurity
- New pre-trained NLP models like GPT-2, BERT can make this approach even more dangerous
- While machine-generated grammar is bad, it is good enough for Twitter
- Abundant personal data on social media platforms can be used for social engineering

Questions?

Thank you for your attention.

Stefan Stein, @SteinsZeit

WARWICK
THE UNIVERSITY OF WARWICK

References

Video of the talk can be found here. Original authors' Twitter:

- John Seymour @_delta_zero
- Philip Tully @phtully

Original slides:

Seymour, J, and P Tully. 2016. "Weaponizing Data Science for Social Engineering: Automated E2e Spear Phishing on Twitter." https://www.blackhat.com/docs/us-16/materials/us-16-Seymour-Tully-Weaponizing-Data-Science-For-Social-Engineering-Automated-E2E-Spear-Phishing-On-Twitter.pdf.