Exercise 2

Let’s Communicate II | NLP Exercise | Sharon Ku

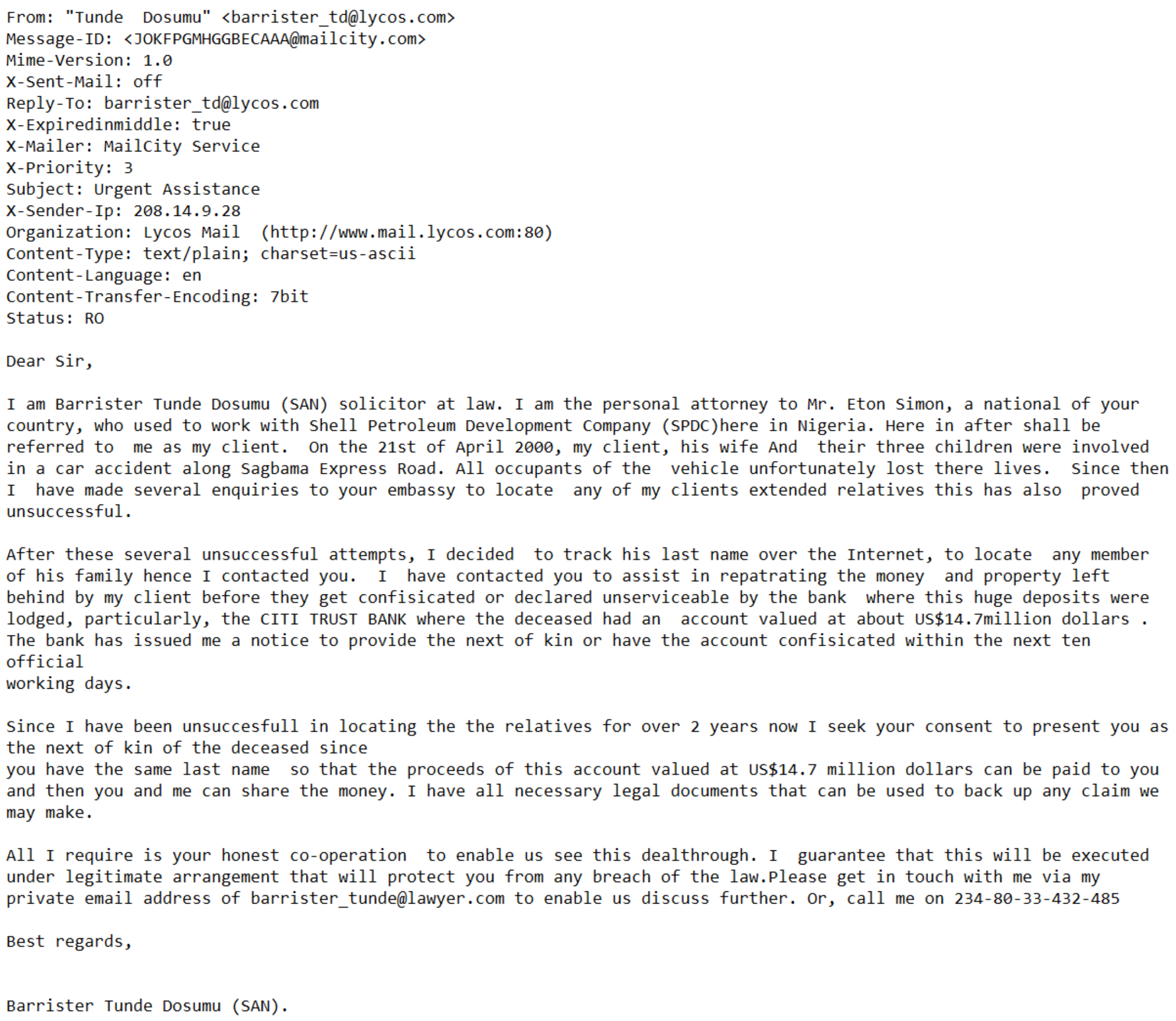
The Corpus

My corpus is composed of data from the CLAIR collection of more than 2500 "Nigerian" fraud emails, gathered from 1998 to 2007. I did not have any particular datasets in mind when tackling this project, so after a quick search of .txt files on Kaggle, I came across this dataset that piqued my interest because I love engaging in conversations with my scammers in my free time. I’m fascinated by how the scammers approach extorting money from me and observing their assumptions of me (I always take on the persona of an old man).

I was interested in seeing what words, phrases, or tactics were used in these “Nigerian” fraud emails. This dataset was helpful because it contained information such as the message sender, the date, the email subject line, and the message. I also thought that the theme of scamming can lead to unique visualizations.

Of the 2500 fraud emails, I randomly chose a sample of 12 emails due to limited time. All the emails were located in a single text file, so I had to copy-paste the emails into individual files to explore the NLP techniques.

Below is a sample email:



```
From: "Tunde  Dosumu" <barrister_td@lycos.com>
Message-ID: <JOKFPGMHGGBECAAA@mailcity.com>
Mime-Version: 1.0
X-Sent-Mail: off
Reply-To: barrister_td@lycos.com
X-Expiredinmiddle: true
X-Mailer: MailCity Service
X-Priority: 3
Subject: Urgent Assistance
X-Sender-Ip: 208.14.9.28
Organization: Lycos Mail  (http://www.mail.lycos.com:80)
Content-Type: text/plain; charset=us-ascii
Content-Language: en
Content-Transfer-Encoding: 7bit
Status: RO

Dear Sir,

I am Barrister Tunde Dosumu (SAN) solicitor at law. I am the personal attorney to Mr. Eton Simon, a national of your
country, who used to work with Shell Petroleum Development Company (SPDC)here in Nigeria. Here in after shall be
referred to  me as my client.  On the 21st of April 2000, my client, his wife And  their three children were involved
in a car accident along Sagbama Express Road. All occupants of the  vehicle unfortunately lost there lives.  Since then
I  have made several enquiries to your embassy to locate  any of my clients extended relatives this has also  proved
unsuccessful.

After these several unsuccessful attempts, I decided  to track his last name over the Internet, to locate  any member
of his family hence I contacted you.  I  have contacted you to assist in repatrating the money  and property left
behind by my client before they get confisicated or declared unserviceable by the bank  where this huge deposits were
lodged, particularly, the CITI TRUST BANK where the deceased had an  account valued at about US$14.7million dollars .
The bank has issued me a notice to provide the next of kin or have the account confisicated within the next ten
official
working days.

Since I have been unsuccesfull in locating the the relatives for over 2 years now I seek your consent to present you as
the next of kin of the deceased since
you have the same last name  so that the proceeds of this account valued at US$14.7 million dollars can be paid to you
and then you and me can share the money. I have all necessary legal documents that can be used to back up any claim we
may make.

All I require is your honest co-operation  to enable us see this dealthrough. I  guarantee that this will be executed
under legitimate arrangement that will protect you from any breach of the law.Please get in touch with me via my
private email address of barrister_tunde@lawyer.com to enable us discuss further. Or, call me on 234-80-33-432-485

Best regards,


Barrister Tunde Dosumu (SAN).
```

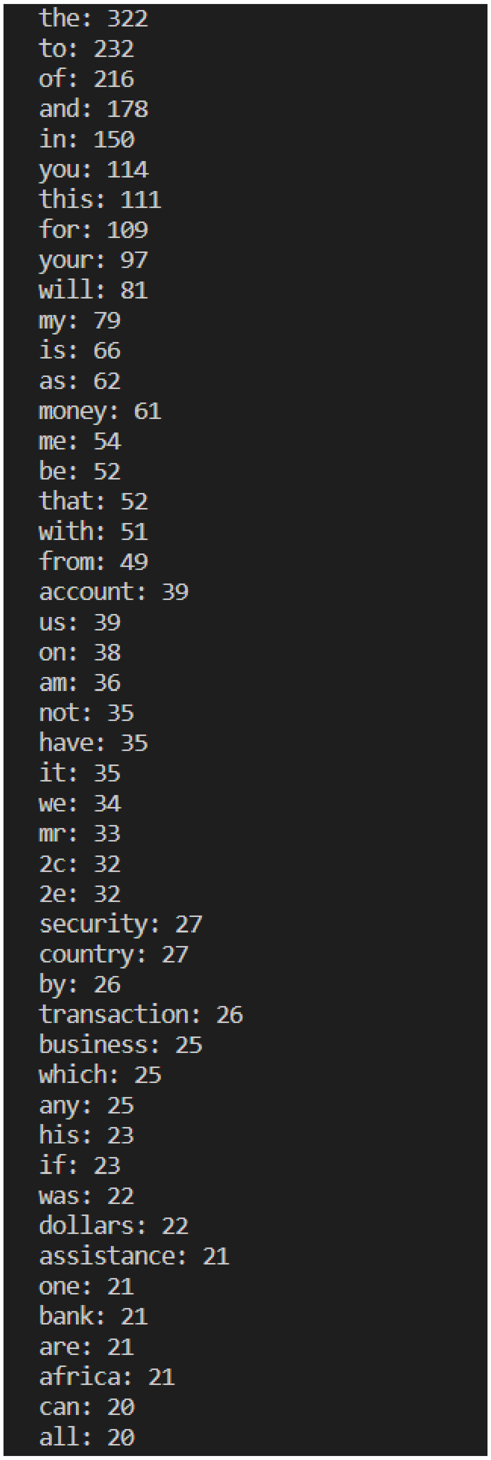
NLP Explorations

1 → Word frequencies, tf-idf

- Files: fraudWordCount.js, fraudTFIDF.js
- Results files: word-count-results.txt, tfidf-results-w-count.txt, tfidf-results-only-keywords.txt

In **fraudWordCount.js**, I calculated the most frequent words across the 12 emails.

Screenshot of the top results:



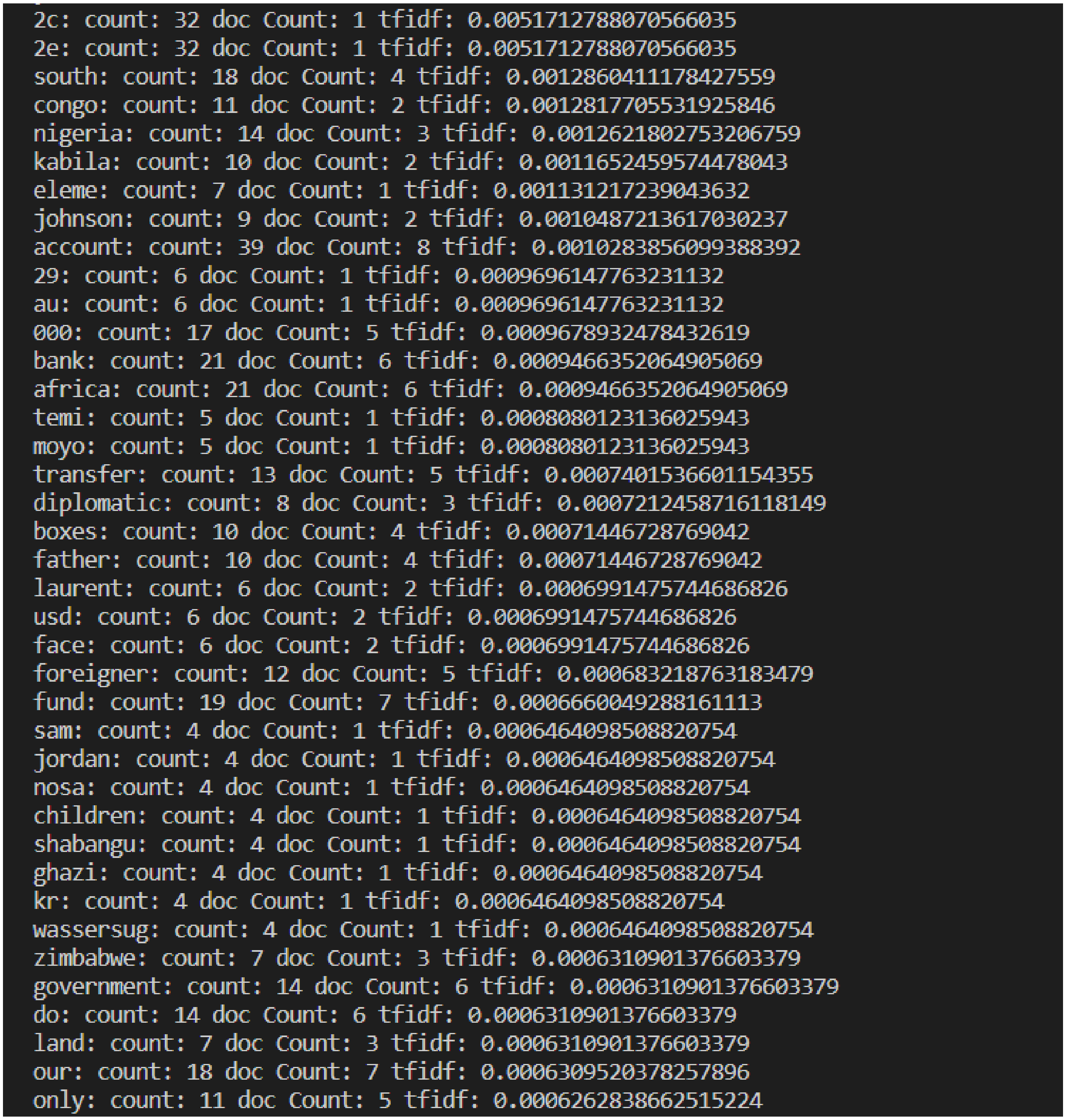
```
the: 322
to: 232
of: 216
and: 178
in: 150
you: 114
this: 111
for: 109
your: 97
will: 81
my: 79
is: 66
as: 62
money: 61
me: 54
be: 52
that: 52
with: 51
from: 49
account: 39
us: 39
on: 38
am: 36
not: 35
have: 35
it: 35
we: 34
mr: 33
2c: 32
2e: 32
security: 27
country: 27
by: 26
transaction: 26
business: 25
which: 25
any: 25
his: 23
if: 23
was: 22
dollars: 22
assistance: 21
one: 21
bank: 21
are: 21
africa: 21
can: 20
all: 20
```

Results are stored in **word-count-results.txt**.

If I disregard what I consider to be common, unimportant words (like prepositions, determinants, and adverbs), I see that the more interesting words are: money, account, security, country, transaction, business, dollars, assistance, bank, and africa. Many of these words relate to money and business. There are also frequent references to a male person with “mr” and “his.”

Next, I calculated the tf-idf values of the 12 emails in **fraudTFIDF.js**.

Screenshot of part of the results, ranked by highest tf-idf values:



```
2c: count: 32 doc Count: 1 tfidf: 0.0051712788070566035
2e: count: 32 doc Count: 1 tfidf: 0.0051712788070566035
south: count: 18 doc Count: 4 tfidf: 0.0012860411178427559
congo: count: 11 doc Count: 2 tfidf: 0.0012817705531925846
nigeria: count: 14 doc Count: 3 tfidf: 0.0012621802753206759
kabila: count: 10 doc Count: 2 tfidf: 0.0011652459574478043
eleme: count: 7 doc Count: 1 tfidf: 0.001131217239043632
johnson: count: 9 doc Count: 2 tfidf: 0.0010487213617030237
account: count: 39 doc Count: 8 tfidf: 0.0010283856099388392
29: count: 6 doc Count: 1 tfidf: 0.0009696147763231132
au: count: 6 doc Count: 1 tfidf: 0.0009696147763231132
000: count: 17 doc Count: 5 tfidf: 0.0009678932478432619
bank: count: 21 doc Count: 6 tfidf: 0.0009466352064905069
africa: count: 21 doc Count: 6 tfidf: 0.0009466352064905069
temi: count: 5 doc Count: 1 tfidf: 0.0008080123136025943
moyo: count: 5 doc Count: 1 tfidf: 0.0008080123136025943
transfer: count: 13 doc Count: 5 tfidf: 0.0007401536601154355
diplomatic: count: 8 doc Count: 3 tfidf: 0.0007212458716118149
boxes: count: 10 doc Count: 4 tfidf: 0.00071446728769042
father: count: 10 doc Count: 4 tfidf: 0.00071446728769042
laurent: count: 6 doc Count: 2 tfidf: 0.0006991475744686826
usd: count: 6 doc Count: 2 tfidf: 0.0006991475744686826
face: count: 6 doc Count: 2 tfidf: 0.0006991475744686826
foreigner: count: 12 doc Count: 5 tfidf: 0.000683218763183479
fund: count: 19 doc Count: 7 tfidf: 0.0006660049288161113
sam: count: 4 doc Count: 1 tfidf: 0.0006464098508820754
jordan: count: 4 doc Count: 1 tfidf: 0.0006464098508820754
nosa: count: 4 doc Count: 1 tfidf: 0.0006464098508820754
children: count: 4 doc Count: 1 tfidf: 0.0006464098508820754
shabangu: count: 4 doc Count: 1 tfidf: 0.0006464098508820754
ghazi: count: 4 doc Count: 1 tfidf: 0.0006464098508820754
kr: count: 4 doc Count: 1 tfidf: 0.0006464098508820754
wassersug: count: 4 doc Count: 1 tfidf: 0.0006464098508820754
zimbabwe: count: 7 doc Count: 3 tfidf: 0.0006310901376603379
government: count: 14 doc Count: 6 tfidf: 0.0006310901376603379
do: count: 14 doc Count: 6 tfidf: 0.0006310901376603379
land: count: 7 doc Count: 3 tfidf: 0.0006310901376603379
our: count: 18 doc Count: 7 tfidf: 0.0006309520378257896
only: count: 11 doc Count: 5 tfidf: 0.0006262838662515224
```

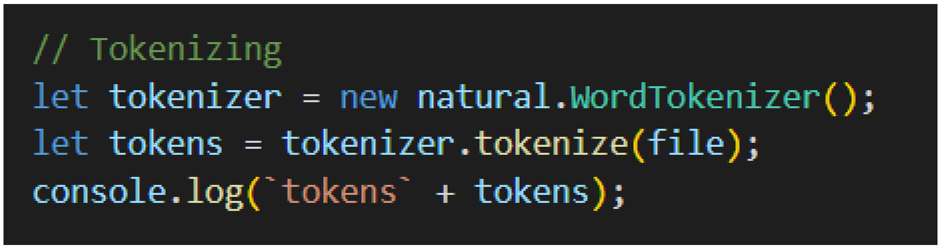
The results are stored in **tfidf-results-w-count.js**. Paying no attention to words that only appear in one document, I notice that the top words are: south, congo, nigeria, kabila, johnson, account, 000, bank, africa, transfer, diplomatic, boxes, father, laurent, usd, face, foreigner, etc. And if I investigate words that do not relate to Africa or location, the top words are: johnson, account, 000, bank, transfer, diplomatic, boxes, father, laurent, usd, face, foreigner. This suggests that most of the time, the scammer uses money and politics in the context of their emails.

I stored only the keys in a .txt file for later use: **tfidf-results-only-keywords.txt**.

2 → Tokenize, Stemming

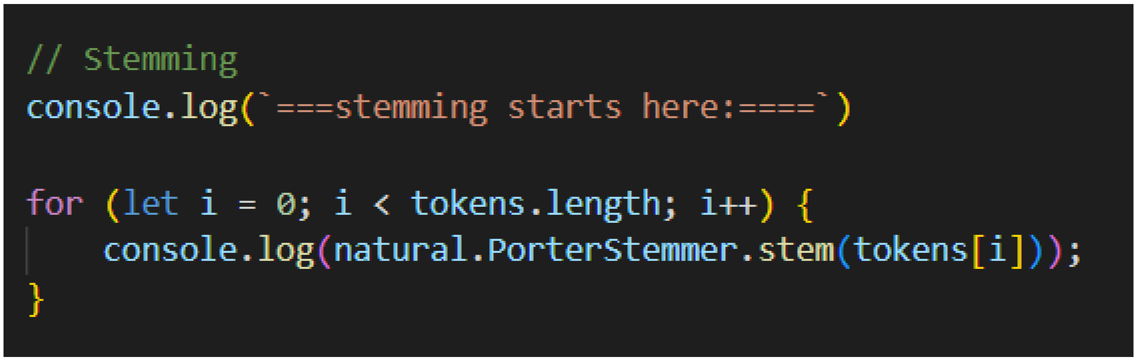
- File: fraudNatural.js
- Result files: stemming-results.txt

Tokenizing:



```
// Tokenizing
let tokenizer = new natural.WordTokenizer();
let tokens = tokenizer.tokenize(file);
console.log(`tokens` + tokens);
```

Stemming:



```
// Stemming
console.log(`===stemming starts here:====`)

for (let i = 0; i < tokens.length; i++) {
    console.log(natural.PorterStemmer.stem(tokens[i]));
}
```

Stemming results were stored in: **stemming-results.txt**. These weren't used in my final visualization since I decided to use verbs and not nouns.

3 → N-grams, POS Tags

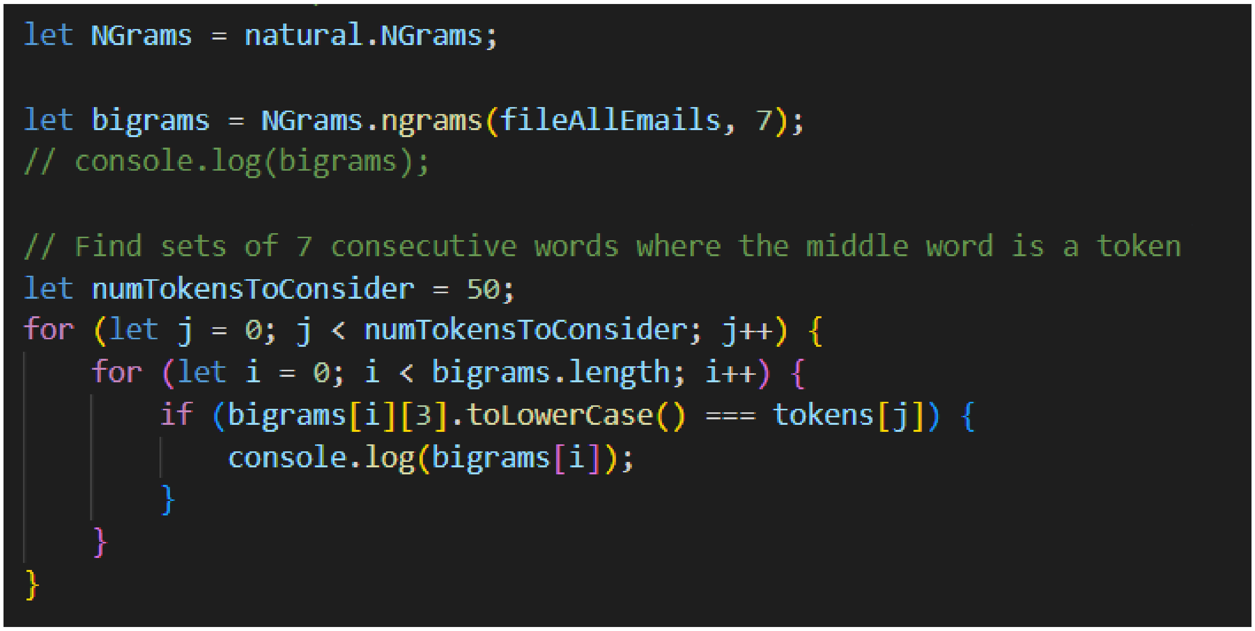
- File: fraudNatural.js
- Result files: tfidf-results-only-keywords-more-than-once-only-verbs.txt, pos-tags-top-results.txt, phrases-with-top-verbs.txt

N-grams:

From the top tf-idf keys, I removed those that only appear in one document, storing them in a file called **tfidf-results-only-keywords-more-than-once.txt**. I've only checked the first 250 results.

I tokenized them and taking into consideration the first 50 tokens, I found the n-grams of 7 words, in which the token is the middle word so that I can get the context around it.

Code:

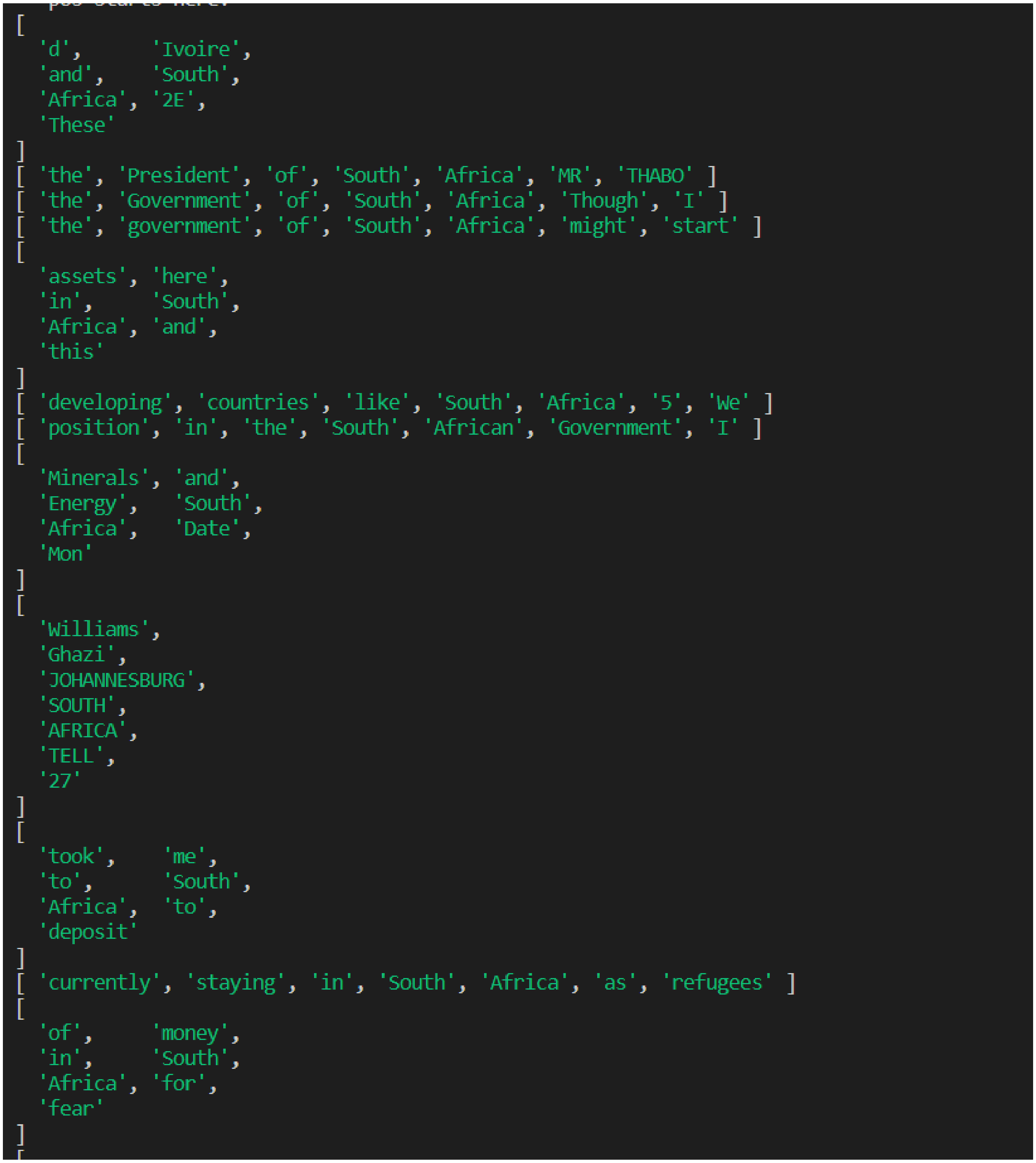


```javascript
let NGrams = natural.NGrams;

let bigrams = NGrams.ngrams(fileAllEmails, 7);
// console.log(bigrams);

// Find sets of 7 consecutive words where the middle word is a token
let numTokensToConsider = 50;
for (let j = 0; j < numTokensToConsider; j++) {
    for (let i = 0; i < bigrams.length; i++) {
        if (bigrams[i][3].toLowerCase() === tokens[j]) {
            console.log(bigrams[i]);
        }
    }
}
```

Screenshot of result:



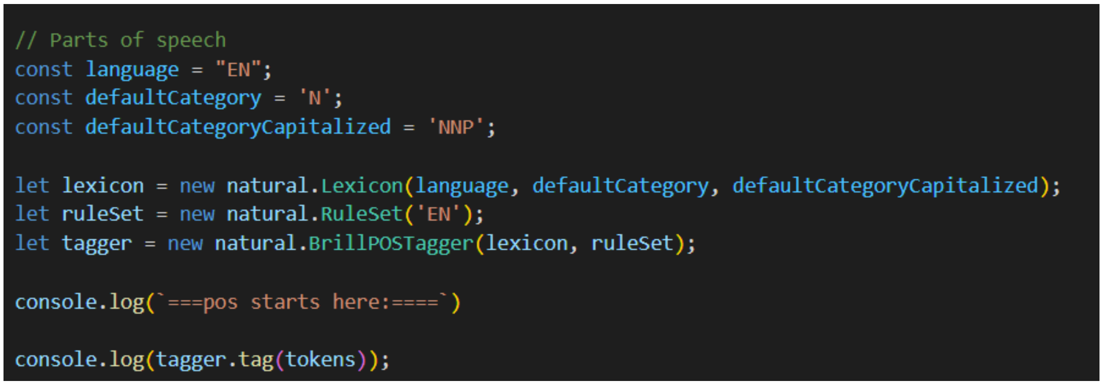
```
[
  'd',      'Ivoire',
  'and',    'South',
  'Africa', '2E',
  'These'
]
[ 'the', 'President', 'of', 'South', 'Africa', 'MR', 'THABO' ]
[ 'the', 'Government', 'of', 'South', 'Africa', 'Though', 'I' ]
[ 'the', 'government', 'of', 'South', 'Africa', 'might', 'start' ]
[
  'assets', 'here',
  'in',     'South',
  'Africa', 'and',
  'this'
]
[ 'developing', 'countries', 'like', 'South', 'Africa', '5', 'We' ]
[ 'position', 'in', 'the', 'South', 'African', 'Government', 'I' ]
[
  'Minerals', 'and',
  'Energy',   'South',
  'Africa',   'Date',
  'Mon'
]
[
  'Williams',
  'Ghazi',
  'JOHANNESBURG',
  'SOUTH',
  'AFRICA',
  'TELL',
  '27'
]
[
  'took',    'me',
  'to',      'South',
  'Africa',  'to',
  'deposit'
]
[ 'currently', 'staying', 'in', 'South', 'Africa', 'as', 'refugees' ]
[
  'of',     'money',
  'in',     'South',
  'Africa', 'for',
  'fear'
]
[
```

However, I was more intrigued by the phases that contained verbs, in which the scammer is describing a made-up situation or ordering the message receiver to do something.

POS Tags:

Therefore, I turned to POS tags annotation to grab all the verbs in my top list of TFIDF keys.

Code:



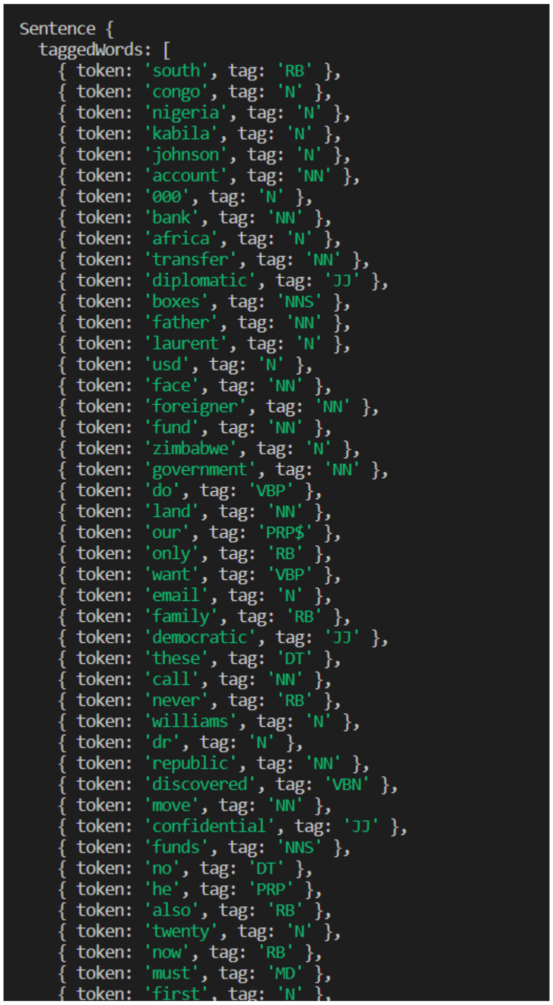
```
// Parts of speech
const language = "EN";
const defaultCategory = 'N';
const defaultCategoryCapitalized = 'NNP';

let lexicon = new natural.Lexicon(language, defaultCategory, defaultCategoryCapitalized);
let ruleSet = new natural.RuleSet('EN');
let tagger = new natural.BrillPOSTagger(lexicon, ruleSet);

console.log(`===pos starts here:====`)

console.log(tagger.tag(tokens));
```

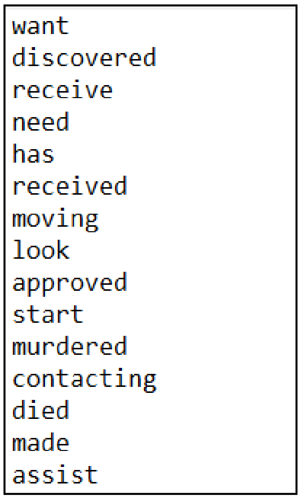
Screenshot of top results:



```
Sentence {
  taggedWords: [
    { token: 'south', tag: 'RB' },
    { token: 'congo', tag: 'N' },
    { token: 'nigeria', tag: 'N' },
    { token: 'kabila', tag: 'N' },
    { token: 'johnson', tag: 'N' },
    { token: 'account', tag: 'NN' },
    { token: '000', tag: 'N' },
    { token: 'bank', tag: 'NN' },
    { token: 'africa', tag: 'N' },
    { token: 'transfer', tag: 'NN' },
    { token: 'diplomatic', tag: 'JJ' },
    { token: 'boxes', tag: 'NNS' },
    { token: 'father', tag: 'NN' },
    { token: 'laurent', tag: 'N' },
    { token: 'usd', tag: 'N' },
    { token: 'face', tag: 'NN' },
    { token: 'foreigner', tag: 'NN' },
    { token: 'fund', tag: 'NN' },
    { token: 'zimbabwe', tag: 'N' },
    { token: 'government', tag: 'NN' },
    { token: 'do', tag: 'VBP' },
    { token: 'land', tag: 'NN' },
    { token: 'our', tag: 'PRP$' },
    { token: 'only', tag: 'RB' },
    { token: 'want', tag: 'VBP' },
    { token: 'email', tag: 'N' },
    { token: 'family', tag: 'RB' },
    { token: 'democratic', tag: 'JJ' },
    { token: 'these', tag: 'DT' },
    { token: 'call', tag: 'NN' },
    { token: 'never', tag: 'RB' },
    { token: 'williams', tag: 'N' },
    { token: 'dr', tag: 'N' },
    { token: 'republic', tag: 'NN' },
    { token: 'discovered', tag: 'VBN' },
    { token: 'move', tag: 'NN' },
    { token: 'confidential', tag: 'JJ' },
    { token: 'funds', tag: 'NNS' },
    { token: 'no', tag: 'DT' },
    { token: 'he', tag: 'PRP' },
    { token: 'also', tag: 'RB' },
    { token: 'twenty', tag: 'N' },
    { token: 'now', tag: 'RB' },
    { token: 'must', tag: 'MD' },
    { token: 'first', tag: 'N' },
```

The top results were stored in: **pos-tags-top-results.txt**

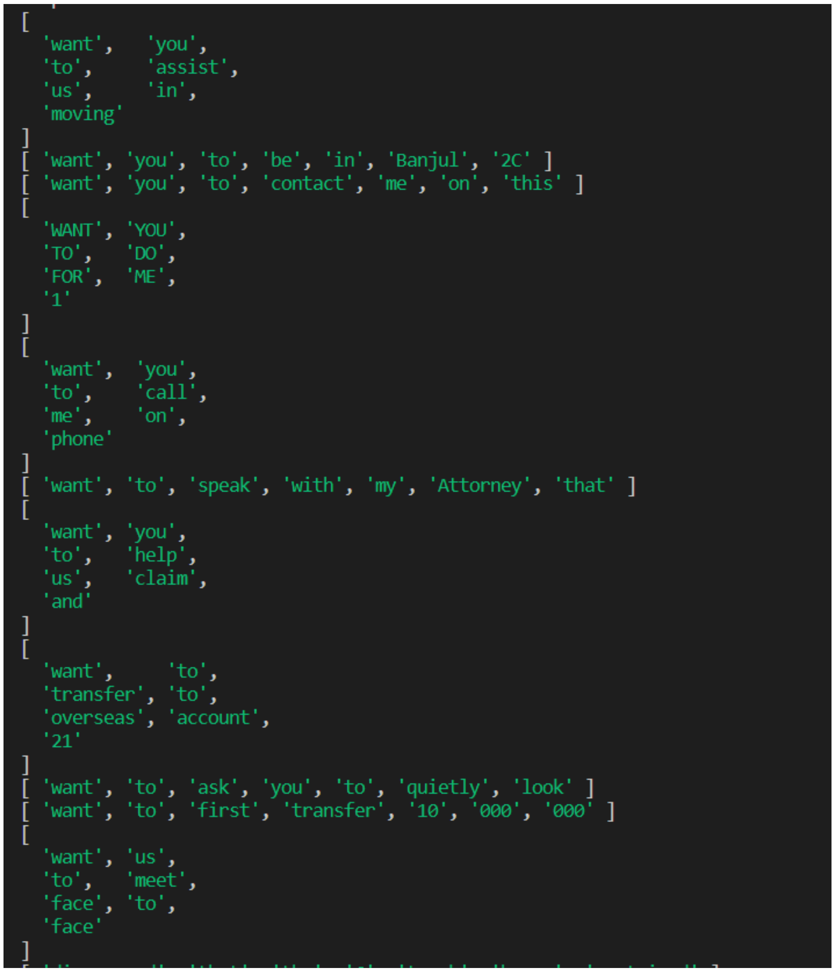
This information was the most helpful as it showed what words were classified as verbs. I logged the top 15 verbs into
tfidf-results-only-keywords-more-than-once-only-verbs.txt
(These are the top TFIDF results, where the keywords appear in more than one document, and that are only verbs.)



```
want
discovered
receive
need
has
received
moving
look
approved
start
murdered
contacting
died
made
assist
```

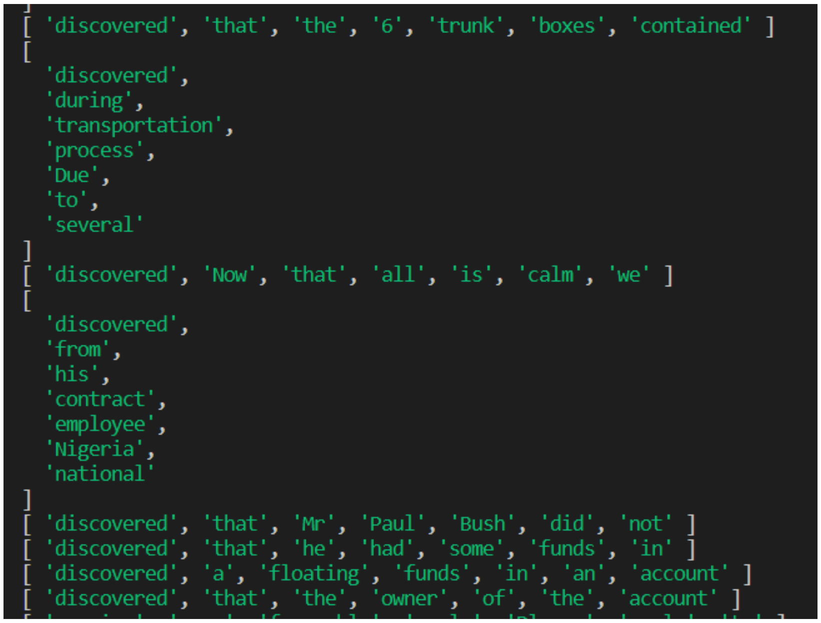
Back to using n-grams, I search for the set of 10 consecutive words in which the verb is the third word. This will allow me to see the subject of the verb and the object following the verb. I logged the result in **phrases-with-top-verbs.txt**.

The results revealed the stories that scammers tended to craft to manipulate the reader. The fact that the verb “want” takes the top position indicates that the scammer often issues an order, such as in the examples below:



```
[
  'want',   'you',
  'to',     'assist',
  'us',     'in',
  'moving'
]
[ 'want', 'you', 'to', 'be', 'in', 'Banjul', '2C' ]
[ 'want', 'you', 'to', 'contact', 'me', 'on', 'this' ]
[
  'WANT', 'YOU',
  'TO',   'DO',
  'FOR',  'ME',
  '1'
]
[
  'want',  'you',
  'to',    'call',
  'me',    'on',
  'phone'
]
[ 'want', 'to', 'speak', 'with', 'my', 'Attorney', 'that' ]
[
  'want', 'you',
  'to',   'help',
  'us',   'claim',
  'and'
]
[
  'want',     'to',
  'transfer', 'to',
  'overseas', 'account',
  '21'
]
[ 'want', 'to', 'ask', 'you', 'to', 'quietly', 'look' ]
[ 'want', 'to', 'first', 'transfer', '10', '000', '000' ]
[
  'want', 'us',
  'to',   'meet',
  'face', 'to',
  'face'
]
```

The verb “discovered” is commonly used in the fictional stories that the scammer makes up to explain that a problem occurred:



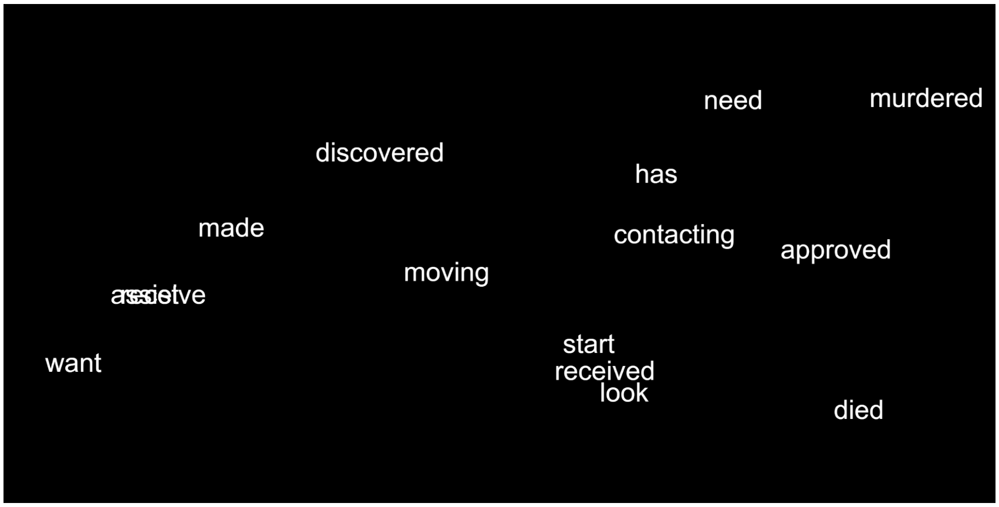
```
[ 'discovered', 'that', 'the', '6', 'trunk', 'boxes', 'contained' ]
[
  'discovered',
  'during',
  'transportation',
  'process',
  'Due',
  'to',
  'several'
]
[ 'discovered', 'Now', 'that', 'all', 'is', 'calm', 'we' ]
[
  'discovered',
  'from',
  'his',
  'contract',
  'employee',
  'Nigeria',
  'national'
]
[ 'discovered', 'that', 'Mr', 'Paul', 'Bush', 'did', 'not' ]
[ 'discovered', 'that', 'he', 'had', 'some', 'funds', 'in' ]
[ 'discovered', 'a', 'floating', 'funds', 'in', 'an', 'account' ]
[ 'discovered', 'that', 'the', 'owner', 'of', 'the', 'account' ]
```

I also noticed there are associations to death, like “murdered” and “died.” Inheritance issues were the way to fool victims of scam messages.

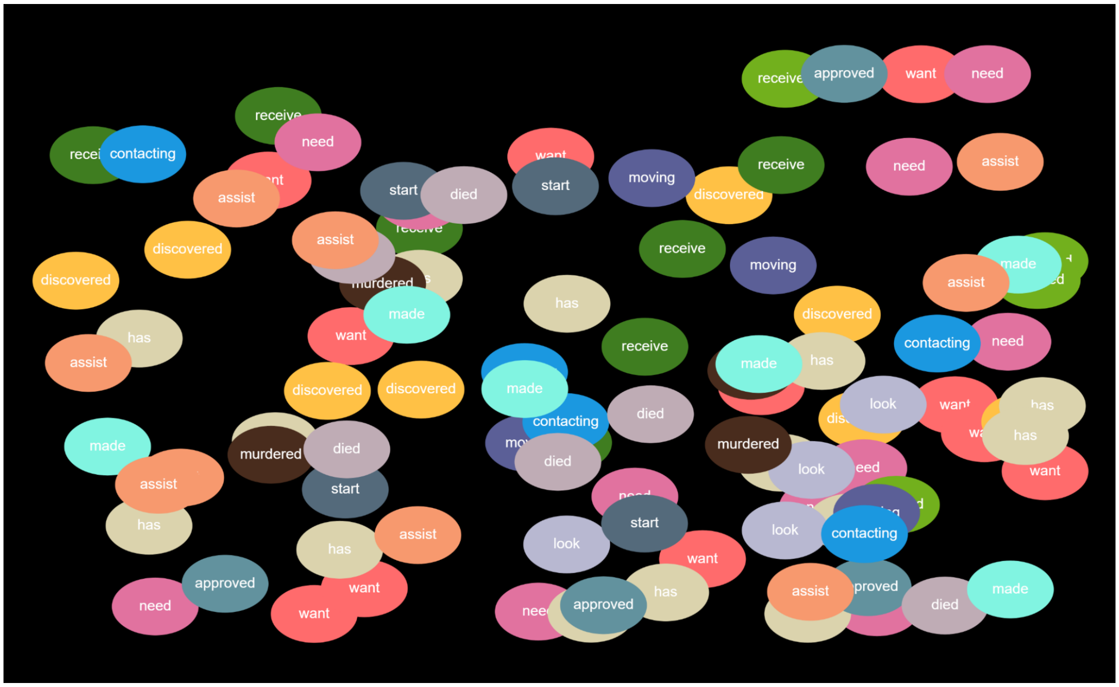
Visualization

Progress

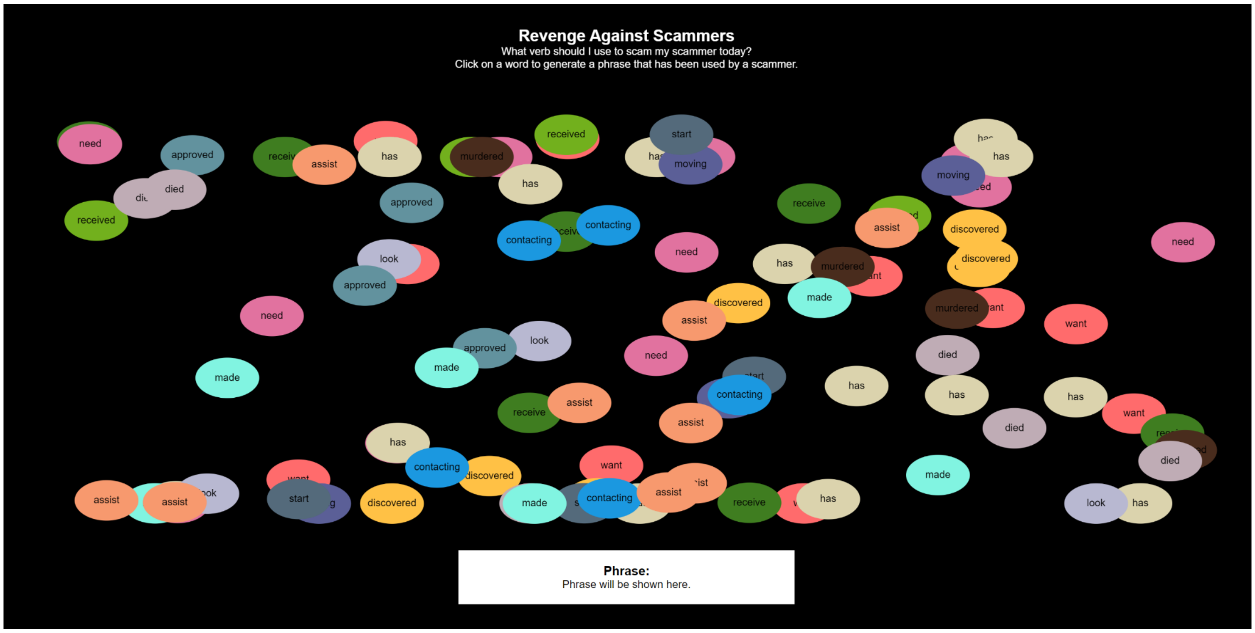
I began by displaying the 15 verbs with the highest TFIDF values and that appear in at least one document. They are floating randomly around the window.



I wanted the ability for people to see the phrase of the verb and the number of phrases associated with the verb. So I color-coded each verb, assigning them with a random colored circle. This shows how many phrases there are for each verb.



Next, I wanted to make the theme of email scamming more obvious. I recalled my initial reason for being drawn to the scam email dataset. So I created a story and purpose behind this visualization: it's intended to give inspiration for phrases to send to one's scammer. Therefore, when the user clicks on a verb, it will grab the phrase attached to that verb and display it in the white box.



Revenge Against Scammers

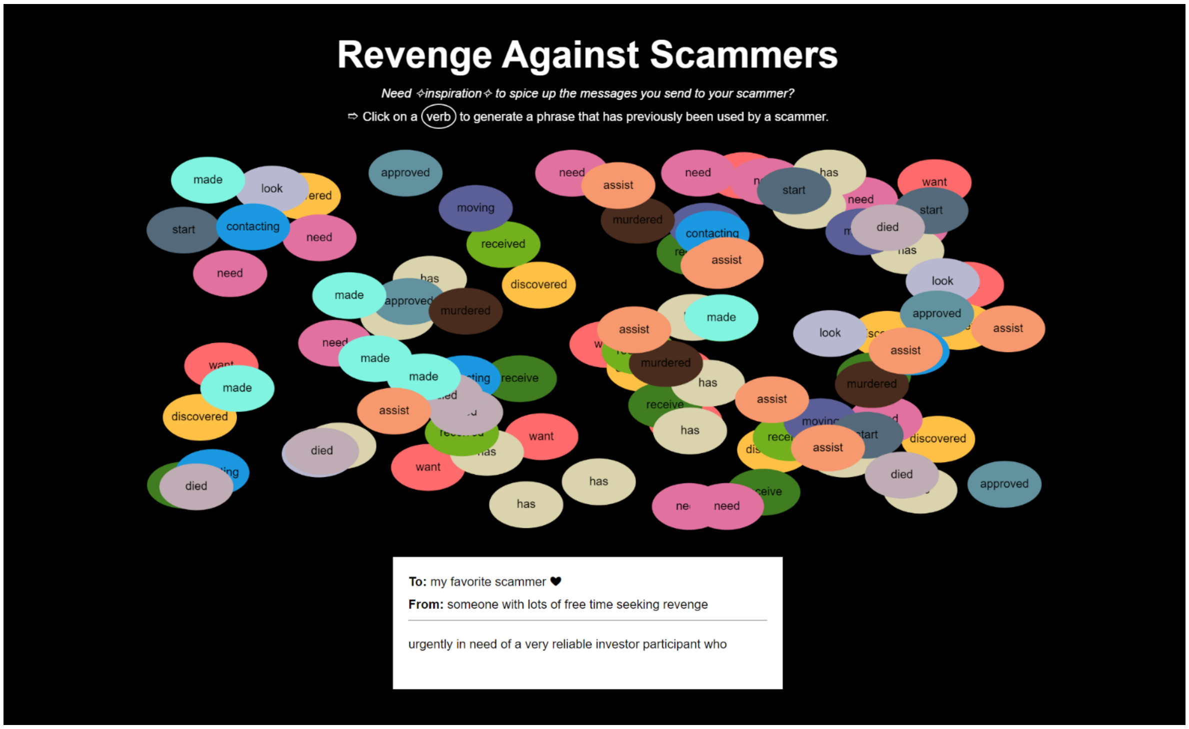
What verb should I use to scam my scammer today?

Click on a word to generate a phrase that has been used by a scammer.

Phrase:

Phrase will be shown here.

I added a message box at the bottom to clarify the context of using the phrase. I made the instructions clearer.



Revenge Against Scammers

Need ✧inspiration✧ to spice up the messages you send to your scammer?

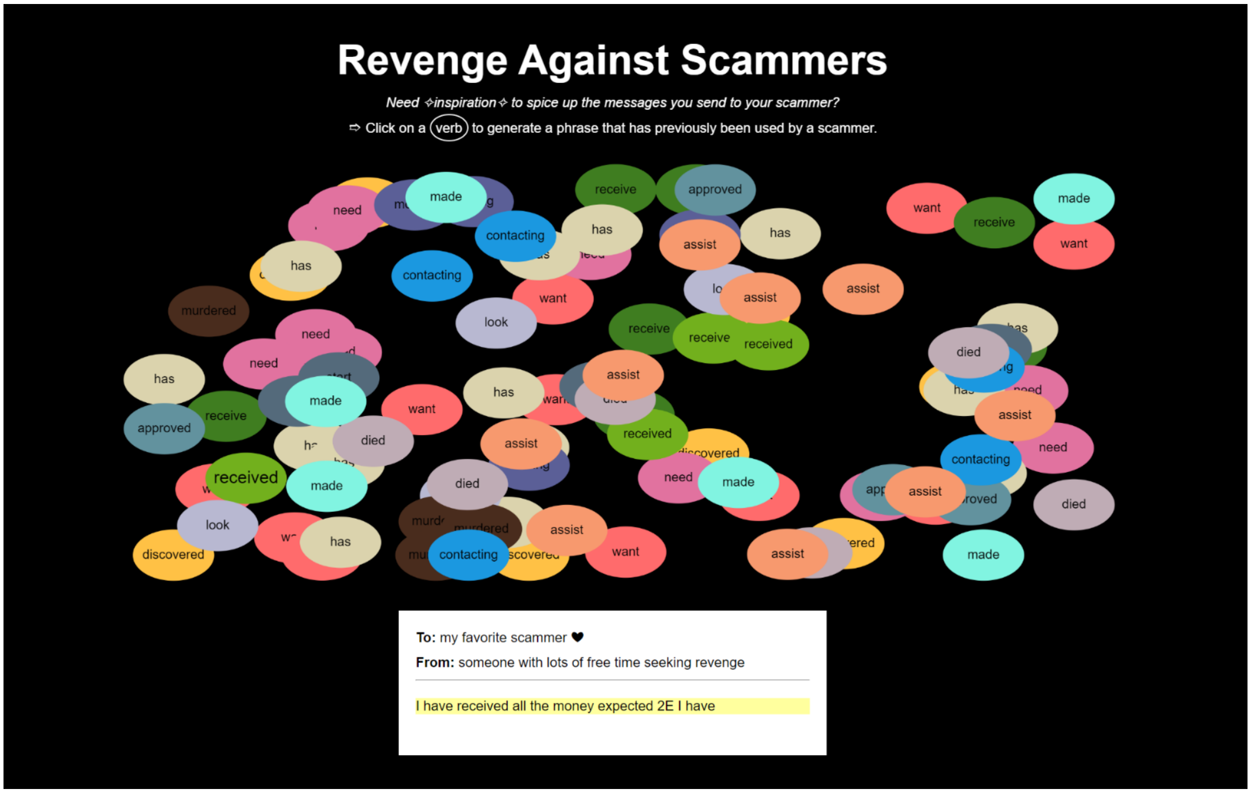
➩ Click on a (verb) to generate a phrase that has previously been used by a scammer.

To: my favorite scammer ❤

From: someone with lots of free time seeking revenge

urgently in need of a very reliable investor participant who

Finally, to make it obvious that the phrase text is updating, I add a highlight effect every time a verb is clicked.



Revenge Against Scammers

Need ✧inspiration✧ to spice up the messages you send to your scammer?

⇨ Click on a (verb) to generate a phrase that has previously been used by a scammer.

To: my favorite scammer ♥

From: someone with lots of free time seeking revenge

I have received all the money expected 2E I have

Final Comments

The completed visualization is a comedic approach to show the data gathered from the dataset of 12 fraud emails. It shows the 15 verbs with highest TFIDDF values. By labelling them with colors, the user can see how often the verb were used in the emails. And instead of overwhelming the user with all the phrases, I allowed them to click on a verb to see what phrases are behind them. The visualization is thus functional as well since it gives the users inspiration if they desire to send messages back to their scammers.

Sources

Radev, D. (2008), CLAIR collection of fraud email, ACL Data and Code Repository, ADCR2008T001, http://aclweb.org/aclwiki