

# What's NOT in a name?

Mike McKenna [mgm.globalization\(at\)gmail.com](mailto:mgm.globalization(at)gmail.com)

v.1.2

October, 2017



## Problem

- Money Laundering
- EU – 4<sup>th</sup> Anti-Money Laundering Directive
- Customer due-diligence
- Avoid anonymous or fake names



## Examples

i, ramsiin 00 luvaluva, ramjay 81, rami tqwert, raluca zq  
om, rain qwertyy, rai2722, rahul 1u sharma, rafiq ch2s ah  
1 beriketaki, rafael molina alcontin 2010, radwan nasdf d  
achel marie webber 00, rachael d039eath, rabia ay ssjk 10  
tfgluizn, qwertz yxcvb, qwertz veinteuno, qwertz mars, qw  
z bontrose, qwertyy dsfdyutctfreutqsj, qwertyvahn stoney,  
egeew, qwertyuiopasdfghjklzxcvbnmhi qwertyuioplaksjdhfg,  
cxz, qwertyuiop qwertyuiopas, qwertyuiop qwertyuiiusfghjd  
, qwertyuiop poiuytrewq qpwoeiruty, qwertyuiop keyboardis  
rewq, qwertyu sayo, qwertyu asdfghjkl si, qwertytwo fourse  
qwertys gillea, qwertyrohitafggdfgdf zxcvbm, qwertyq qaz  
rm, qwertynan zult, qwertyfive farm, qwertyfarmtwo farm,  
rm, qwertyfarmone farm, qwertyfarmfour farmer, qwertyeigh  
ey, qwertycat mrcat, qwertyasdf zxc, qwertya carasky, qwe  
qwertu dfgh, qwertyy loltty, qwertsix farm, qwerts sy, qw  
qwertyrt wertyu, qwertyq ahmad, qwertyopus cprules, qwertye  
er, qwertythw kah wai, qwertyer daniel, qwertyteen asesoris,  
qwertye asdfghjklzxcvbnm, qwertye asdfghjklzxcvbnm, qwertye asdfghjklzxcvbnm, qwertye asdfghjklzxcvbnm



## Ideas to find gibberish

- NLP and probability
  - Natural language processing
  - Generated probability
- Issue: each country and language has different aspects
  - Accepted character ranges
  - Common names
  - Different keyboards to bang on

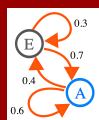


## Training data



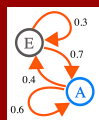
## Training data

- Need *names* not free text
- Regional census databases
- Real estate and property data
- Names, hacker databases



## How to train probability?

- NLP – Natural Language Processing?
- Markov chains
  - Probability of state transitions
  - Square matrix: ABC x ABC
  - $N^2$  data size
  - Simple to implement



## Markov chains

**“MARY”**

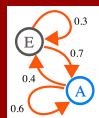
$M \rightarrow A \text{ i}[m][a]++ \text{ (1)}$   
 $A \rightarrow R \text{ i}[a][r]++ \text{ (1)}$   
 $R \rightarrow Y \text{ i}[r][y]++ \text{ (1)}$   
 $Y \rightarrow \_ \text{ i}[y][\_ ]++ \text{ (1)}$

**“MARVIN”**

$M \rightarrow A \text{ i}[m][a]++ \text{ (2)}$   
 $A \rightarrow R \text{ i}[a][r]++ \text{ (2)}$   
 $R \rightarrow V \text{ i}[r][v]++ \text{ (1)}$   
 $V \rightarrow I \text{ i}[v][i]++ \text{ (1)}$

**“BARRY”**

$B \rightarrow R \text{ i}[b][r]++ \text{ (1)}$   
 $A \rightarrow R \text{ i}[a][r]++ \text{ (3)}$   
 $R \rightarrow R \text{ i}[r][r]++ \text{ (1)}$   
 $R \rightarrow Y \text{ i}[r][y]++ \text{ (2)}$   
 $Y \rightarrow \_ \text{ i}[Y][\_ ]++ \text{ (2)}$

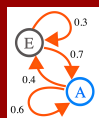


## Markov chains

“JANE”?

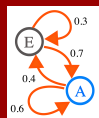
$$\sum_{i=0}^n P(i, i+1) \left\{ \begin{array}{l} P(\_, J) = i[\_][j] \\ P(J, A) = i[j][a] \\ P(A, N) = i[a][n] \\ P(N, E) = i[n][e] \\ P(E, \_) = i[e][\_]\end{array} \right.$$

$n$



## Markov chains - size

- Impractical due to size?
- Unicode  $\approx 10^6$  characters
- Markov space for all Unicode  $\approx 10^{12}$



## Markov chains - size

- If you know the language context
- You can set boundaries
- Reduce to manageable size
- English? -> Latin
- Russian? -> Cyrillic
- Greek? -> Greek



## CLDR to the rescue!

- CLDR defines “exemplarCharacters”
- Each language:  
<https://github.com/unicode-cldr/cldr-misc-modern/blob/master/main/cs/characters.json>
- cldr-misc-modern/.../main/ **<bcp-47 tag>/characters.json**

```
"characters": {
  "exemplarCharacters": "[a á b c ĉ d d e é ě f g h {ch} i í j k l m n ñ o ó p q r ŕ s š t t u ú ů v w x y ý z ž]",
  "auxiliary": "[à â ã ä å æ ç è é ê ë ì í î ï ð ñ ò ó ô õ ö ø ù ü û ü ý]",
  "index": "[A B C Ć D E F G H {CH} I J K L M N O P Q R Ŕ S Š T U V W X Y Z Ž]",
  "punctuation": "[\\- _ , . ; \\: ! ? ' \" , \" ( ) \\[ \\] $ % & / \\\\&]",
```




- ## What's NOT in a Name?



- ## What's NOT in a Name?



## Tuning the alphabets

- Include *exemplar* punctuation for free-text
- Add single not-used character to replace characters outside the alphabet
  - Do not use U+FFFD  because it may legitimately be in text
  - Lesser used chars work – ex: ‘A’



## Training the Stochastic Matrix

### EN

A A'Hern A'Leshia A-Hameed  
 AA AAB AABERG AABY  
 AADLAND AAFEDT  
 AAGAARD AAGARD  
 AAGESEN AAKER AAKHUS  
 AAKRE AALAND AALBERS  
 AALDERINK AALFS  
 AALGAARD AALTO AALUND  
 AAMODT AAMOLD AAMOT  
 AAMOTH AANDERUD  
 AANENSON AANERUD  
 AANESTAD AANONSEN  
 AARANT AARDAL AARDEMA  
 AARDSMA AARESTAD  
 AARHUS AARON AARONS  
 AARONSON AARRIS  
 AARSTAD AARSVOLD ...

### CS

ABASHIN ABAŠIN  
 ABAŠKINA ABAYAZEED  
 ABAZA ABBAWI ABD  
 ELAZIM ABDANK  
 ABŽOLTOVSKÁ ABDEL  
 KARIM ABDEL NOUROVÁ  
 ABDEL-JAWAD  
 ABDELMAGID ABDENNOUR  
 ABDEŠAHMANOVÁ ABDUL  
 ABDUL GHANI ABDUL  
 RAHMAN ABDUL RAHIMOVÁ  
 ABDUL HAMIDOVÁ  
 ABDULANI ABDULSALAM  
 ABDURACHMANOV  
 ABDURACHMANOVA  
 ABDURAZAKOVA  
 ABDÚLOVÁ ABEDOVÁ ...





## Tuning the training data

- Convert to UTF-8
- Normalize with NFC
- Edit for odd strings
  - Punctuation
  - `<e^>`, `<o:>`, accented sequences



## Ideas to find gibberish

- Database of bad strings?
  - All occurrences of “asdf”, “qwerty”, “rumplestiltskins”, ...
- Regex
  - “[qwer]{4,}”, “[asdf]{4,}”, ...
- NLP and probability
  - Generated positive probability ✓
  - Generated negative probability ?

```

02P&Mn)p:dp&:6Bo'+nInFu
h>L<ms8E?'p&:oUr;QKjq>
nD<=D5<.(rWlE'at)[+quf
7ZfqpWVXk6e/\Vs*+A2G3h
OD&[hHeBQa+aQ<.L:>1]j
8EYTe,K.Ap\+B7mIpMXqu
s=jmKLT9qEp\sjXE/95Bo'f
7"4N[nc/V"mJj!Bs7?9's8

```

## Gibberish

- Random key combinations
- Many 4-letter combinations
- 4 fingers? 4 keys!
- "qwertyuiop"? -> "qwer", "wert", ... "uiop"
- Can build pattern generator off keyboard



## Keyboards

;	+	ě	š	č	ř	ž	ý	á	í	é	=	'
q	w	e	r	t	z	u	i	o	p	ú	)	
a	s	d	f	g	h	j	k	l	ú	\$	"	
\	y	x	c	v	b	n	m	.	-			

- <http://www.unicode.org/cldr/charts/31/keyboards/layouts/<lang>.html>
- Defined for various devices and types
  - chromeos
  - windows
  - android
  - OSX
  - ...



# Keyboards

;	+	ě	š	č	ř	ž	ý	á	í	é	'
q	w	e	r	t	z	u	i	o	p	ú	)
a	s	d	f	g	h	j	k	l	ů	š	~
\	y	x	c	v	b	n	m	.	-		

- <http://www.unicode.org/Public/cldr/31.0.1/cldr-keyboards-31.0.1.zip>

```
<keyboard locale="cs-t-k0-windows">
  <version platform="10" number="$Revision: 13265 $" />
  <names>
    <name value="Czech" />
  </names>
  <settings fallback="omit" transformPartial="hide" />
  <keyMap>
    <map iso="E00" to=";" /> <!-- ` -->
    <map iso="E01" to="+" /> <!-- 1 -->
    <map iso="E02" to="ě" /> <!-- 2 -->
    <map iso="E03" to="š" /> <!-- 3 -->
    . . .
    <map iso="D01" to="q" />
    <map iso="D02" to="w" />
    . . .
    <map iso="D06" to="z" /> <!-- Y -->
```



# Keyboards

;	+	ě	š	č	ř	ž	ý	á	í	é	'
q	w	e	r	t	z	u	i	o	p	ú	)
a	s	d	f	g	h	j	k	l	ů	š	~
\	y	x	c	v	b	n	m	.	-		

- Sample negative seeds from keyboard maps
- Remove punctuation not allowed in names

EN  
asdfghjkl  
qwertyuiop  
zxcvbnm

DE  
asdfghjklöä  
azertyuiop  
qsd fghjklmù  
qwertzuiopü  
wxcvbn  
yxcvbnm  
éuèçà  
ßu

CS  
asdfghjklů  
qwertzuiopú  
yxcvbnm  
ěščřžýáíé



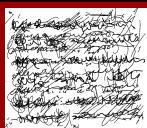
## Generate negative patterns

- All permutations of each 4-key sequence

asdf →  
aaaa aasa      afff  
aaas aass      sfff  
aaad aasd      dfff  
aaaf aasf      ffff

sdfg →  
ssss ssds      gggs  
sssd ssdd      gggd  
sssf ssdf      ggdf  
sssg ssdg      gggs


- Use as negative training set



## Detect gibberish

### ### Detect Gibberish


1. Set high and low positive thresholds
2. Analyze string
  - a. Normalize input string to NFC, LowerCase
  - b. Scan string for characters outside of alphabet
  - c. if outside, then assign to nonChar
  - d. Scan string for positive prob
  - e. Scan string for negative prob
  - f. Subtract negative val from positive val
3. If below positive low threshold then possible gibberish



## Results

Gibberish: <b>false</b> : 0.069090 : ZLATKO	Gibberish: <b>TRUE</b> : 0.016974 : <b>JOHNJOHN</b>
Pos: no : 0.078220	Pos: no : 0.024291
Neg: no : -0.009131	Neg: no : -0.007317
Gibberish: <b>false</b> : 0.109837 : ĚIHÁĚKOVÁ DŮĪMALOVÁ	
Pos: no : 0.120408	
Neg: no : -0.010571	
Gibberish: <b>false</b> : 0.129481 : ŠŽOVÍĚKOVÁ ŠIMÁKOVÁ	
Pos: no : 0.138743	
Neg: no : -0.009262	
	Gibberish: <b>TRUE</b> : -0.080577 : hjkhjkhjk
	Pos: yes : 0.006244
	Neg: yes : -0.086822
Gibberish: <b>TRUE</b> : 0.014407 : GIJOE	Gibberish: <b>TRUE</b> : -0.045154 : qwerty
Pos: no : 0.021526	Pos: no : 0.044269
Neg: no : -0.007119	Neg: yes : -0.089424
Gibberish: <b>TRUE</b> : 0.001914 : Mike.McKenna657	Gibberish: <b>TRUE</b> : -0.082636 : asdfghjk
Pos: yes : 0.013163	Pos: yes : 0.005563
Neg: no : -0.011249	Neg: yes : -0.088199

What's NOT in a Name? Internationalization and Unicode Conference 41 – Santa Clara – October 2017 25



## Results

- High confidence of gibberish detection
  - In One script
  - In one language group
- Found 1000's of fake names in FB data
  - Numbers
  - Bots
  - Keyboard banging

What's NOT in a Name? Internationalization and Unicode Conference 41 – Santa Clara – October 2017 26



## Results

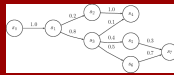
- Demonstrated that CLDR data can be used for:
  - Providing stochastic boundaries for NLP
  - Providing effective random seeds for negative probability through keyboard layouts



## Issues

- Mixed scripts
  - E.g. Cyrillic and Arabic
- CJK
  - Matrix set too large
- Training data
  - Difficulty in getting clean data
- Old data
  - Non-UTF-8, old accent hacks





## References

- **Stochastic matrix**  
In mathematics, a stochastic matrix (also termed probability matrix, transition matrix, substitution matrix, or Markov matrix) is a square matrix used to describe the transitions of a Markov chain. Each of its entries is a nonnegative real number representing a probability. [https://en.wikipedia.org/wiki/Stochastic\\_matrix](https://en.wikipedia.org/wiki/Stochastic_matrix)
- **Typewriters**  
QWERTY refers to the most common form of layout of letters found on the keyboard of a typewriter or computer. The name refers to the first six letters at the top of the board. The initial idea and later development of this design came from one of the first pioneers of the typewriter, Christopher Sholes, who invented the first commercially successful machine. The original layout of letters was in an ABC format, but Sholes found this continually jammed his typewriters. To solve the problem, he asked his brother-in-law, a mathematician, to work out an arrangement that would for most of the time prevent the bars from clashing. Sholes later claimed that this was a highly 'scientific arrangement'. From this the QWERTY idea was evolved in 1873.  
<https://makezine.com/2008/08/15/evolution-of-the-typewrit/>  
<http://www.sciencemuseum.org.uk/onlinestuff/stories/typewriters.aspx>



## References

- **Personal names**  
<https://www.w3.org/International/questions/qa-personal-names>  
[https://www.w3.org/International/wiki/Personal\\_namesW3C](https://www.w3.org/International/wiki/Personal_namesW3C)
- **Norwegian names**  
<https://www.ssb.no/navn>
- **German names**  
<https://daten.berlin.de/datensaetze/liste-der-h%C3%A4ufigen-vornamen-2013>
- **Turkish names**  
<https://gist.github.com/ismailbaskin/1325813> (turkce\_isimler.sql)
- **Greek names**  
<http://www.foundalis.com/grk/EllinikaOnomata.html>
- **French names**  
<https://dataaddict.fr/prenoms/#>
- **Italian names**  
<https://gist.github.com/pdesterlich/2562329> (nomi\_italiani.txt), ASCII  
[https://it.wikipedia.org/wiki/Prenomi\\_italiani\\_\(A-L\)](https://it.wikipedia.org/wiki/Prenomi_italiani_(A-L))  
[https://it.wikipedia.org/wiki/Prenomi\\_italiani\\_\(M-Z\)](https://it.wikipedia.org/wiki/Prenomi_italiani_(M-Z))
- **Vietnamese names**  
<https://inest.hust.edu.vn/danh-sach-sinh-vien>