Fast, Lean, and Accurate: Modeling Password Guessability Using Neural Networks

William Melicher, Blase Ur, Sean Segreti, Saranga Komanduri, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor

Carnegie Mellon
Guessing Methods
Guessing Methods

● John the Ripper

● Hashcat
Guessing Methods

- John the Ripper
  Dictionary word + Rules
- Hashcat
Guessing Methods

- John the Ripper
  - Dictionary word + Rules
- Hashcat
  - password + append 2 digits
Guessing Methods

- John the Ripper
  - Dictionary word + Rules
  - password + append 2 digits
    - password11
    - password12
    - ...

- Hashcat
Guessing Methods

- John the Ripper
- Hashcat
- Markov Models
Guessing Methods

- John the Ripper
- Hashcat
- Markov Models
Guessing Methods

- John the Ripper
- Hashcat
- Markov Models
- PCFGs
Guessing Methods

- John the Ripper
- Hashcat
- Markov Models
- PCFGs

Diagram:

```
L8D2
  /   \
L6S2  ...
     /   \
password12  monkey!!
     \   /  \\
password11  qwerty...
     \ /  \  \\
     ...  ...
```
Guessing Methods

- John the Ripper
- Hashcat
- Markov Models
- PCFGs
Choose a password: **********  
Password strength: Weak  
Minimum of 8 characters in length.

Re-enter password: 
Can we guess more accurately? Quicker? With fewer resources?
Our Approach: Neural Networks

Hello = Здравствуйте

Handwriting Recognition →
Handwriting recognition
Outline: Guessing with Neural Networks

- How to guess passwords with neural networks
- Password guesser design
- Comparison to other guessing methods
- Real-time, in-browser feedback with neural networks
Generating Passwords
Generating Passwords

passw → o or maybe 0 or O or ...
Generating Passwords

Next char is:
A: 3%
B: 1%
C: 0.6%
...
O: 55%
...
Z: 0.01%
0: 20%
1: ...
Generating Passwords

“”

Prob: 100%
Generating Passwords

Next char is:
A: 3%
B: 2%
C: 5%
...
O: 2%
...
Z: 0.2%
0: 1%
1: ...
END: 2%

“”
Prob: 100%
Generating Passwords

Prob: 100%

Next char is:
A: 3%
B: 2%
C: 5%
...
O: 2%
...
Z: 0.2%
0: 1%
1: ...

END: 2%
### Generating Passwords

```
""

Prob: 100%
```

Next char is:

<table>
<thead>
<tr>
<th>Character</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3%</td>
</tr>
<tr>
<td>B</td>
<td>2%</td>
</tr>
<tr>
<td>C</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td>O</td>
<td>2%</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td>Z</td>
<td>0.2%</td>
</tr>
<tr>
<td>0</td>
<td>1%</td>
</tr>
<tr>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>END</td>
<td>2%</td>
</tr>
</tbody>
</table>
Generating Passwords

“C”
Prob: 5%
Generating Passwords

<table>
<thead>
<tr>
<th>Next char is:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10%</td>
</tr>
<tr>
<td>B</td>
<td>1%</td>
</tr>
<tr>
<td>C</td>
<td>4%</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>O</td>
<td>8%</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td>0.02%</td>
</tr>
<tr>
<td>0</td>
<td>3%</td>
</tr>
<tr>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>END</td>
<td>6%</td>
</tr>
</tbody>
</table>
Generating Passwords

“C”
Prob: 5%

Next char is:

A: 10%
B: 1%
C: 4%
...
O: 8%
...
Z: 0.02%
0: 3%
1: ...
END: 6%
Generating Passwords

Next char is:
A: 3%
B: 10%
C: 7%

... 
O: 1%

... 
Z: 0.03%
0: 2%
1: ...
END: 12%

“CA”
Prob: 0.5%
Generating Passwords

“CAB”
Prob: 0.05%

Next char is:
A: 3%
B: 10%
C: 7%
...
O: 1%
...
Z: 0.03%
0: 2%
1: ...
END: 3%
Generating Passwords

“CAB”
Prob: 0.05%

Next char is:
A: 4%
B: 3%
C: 1%
...
O: 2%
...
Z: 0.01%
0: 4%
1: ...
END: 12%
Generating Passwords

“CAB”
Prob: 0.05%

Next char is:

A: 4%
B: 3%
C: 1%
...
O: 2%
...
Z: 0.01%
0: 4%
1: ...
END: 12%
Generating Passwords

“CAB”
Prob: 0.006%
Generating Passwords

CAB - 0.006%
CAC - 0.0042%
ADD1 - 0.002%
CODE - 0.0013%
...

Generating Passwords

CAP - 0.006%
CAC - 0.0042%
ADD1 - 0.002%
CODE - 0.0013%

...
Password Policies: 1class8

1 character class and 8 characters minimum

password123
12345678
monkey99
Password Policies: 4class8

4 character classes and 8 characters minimum

Pa$$w0rd

!Qaz2wsx

Jvj24601!
Password Policies: 1class16

1 character class and 16 characters minimum

123456789123456789
qwertyuiop123456
Monika1234567890
Password Policies: 3class12

3 character class and 12 characters minimum

llamalove123
Mypassword#3
N@rut0_r0ck5
Outline: Guessing with Neural Networks

● How to guess passwords with neural networks

● Password guesser design

● Comparison to other guessing methods

● Real-time, in-browser feedback with neural networks
Design Space
Design Space

- Model size
  
  3MB - Browser
  
  60MB - Limited by GPU
Design Space

- Model size
- Transference learning
Design Space

- Model size
- Transference learning
- Training data

Natural language?
Varying training sets?
Design Space

- Model size
- Transference learning
- Training data
- Model architecture
- Alphabet size
- Password context
Testing Methodology

● Approach: measure # guessed passwords

● Training data: leaked password sets

● Testing data
  ○ MTurk study passwords: 1class8, 4class8, 1class16, 3class12
  ○ Real passwords: 000webhost password leak

● Use Monte-Carlo to estimate guess numbers (Dell’Amico and Filippone CCS ‘15)
Tuning Training
More accurate guessing
More accurate guessing
Transference Learning → More Accurate

15% → 22%
Natural Language Doesn’t Help

![Graph showing the effect of natural language on guessing success](image-url)
Model Size: Larger Is More Accurate
Model Size: Larger Is More Accurate
Model Size: Larger Is More Accurate
Model Size: Larger Is More Accurate

Sometimes
Comparison to Other Approaches
1class8: Comparison

![Graph showing comparison of percent guessed against guesses for Markov PCFG, Hashcat, and JTR.](image)
class8: Neural Networks Guess Better

![Graph showing percent guessed vs guesses](image)

- Neural Markov
- PCFG
- Hashcat
- JTR
1class8: Neural Networks Guess Better
4class8: Neural Networks Guess Better
3class12: Neural Networks Guess Better

![Graph showing percent guessed vs guesses for different methods: MinGuess, Neural, Markov, PCFG, JTR, Hashcat. The graph illustrates how neural networks guess better compared to other methods.](image)
3class12: Neural Networks Guess Better

30% → 45%
Password feedback
Current password feedback: Quick or accurate
Accurate Guessing Methods

100s MB to GBs!
Accurate Guessing Methods

100s MB to GBs!
Accurate Guessing Methods

100s MB to GBs!

Neural networks: 60MB, 3MB
Accurate Guessing Methods

Neural networks: 60MB, 3MB
Accurate Guessing Methods

Hours to days!
Can neural networks give real-time feedback?
Ideal Meter Targets

- Small: < 1MB
- Fast: < 0.1 sec
- JavaScript
- Accurate
Making Meters Small

- Start with small version of neural network
- Quantize parameters of model
- Compress with existing lossless compression methods

850KB < 1MB
Making Meters Fast

- Pre-compute inexact mapping from prob → guess number
- Cache intermediate results
- Run on separate thread

17 ms  < 0.1 sec
Meter Accuracy
Meter Accuracy

Percent guessed

0%

25%

50%

75%

100%

10^1 10^4 10^7 10^{10} 10^{13} 10^{16} 10^{19} 10^{22} 10^{25}

Guesses

Server
Meter Accuracy
Meter Accuracy

Percent guessed

0% 25% 50% 75% 100%

Guesses

$10^1$ $10^4$ $10^7$ $10^{10}$ $10^{13}$ $10^{16}$ $10^{19}$ $10^{22}$ $10^{25}$

Server

Browser
Modeling Passwords Using Neural Networks

- Neural networks guess passwords accurately
- Can be made small and fast for client-side feedback

github.com/cupslab

William Melicher, Blase Ur, Sean M. Segreti, Saranga Komanduri, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor

Carnegie Mellon