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**
- **Hashcat**

Dictionary word + Rules

Guessing Methods

- **John the Ripper**
- **Hashcat**

Dictionary word + Rules

`password` + append 2 digits

Guessing Methods

- **John the Ripper**
- **Hashcat**

Dictionary word + Rules

`password` + append 2 digits

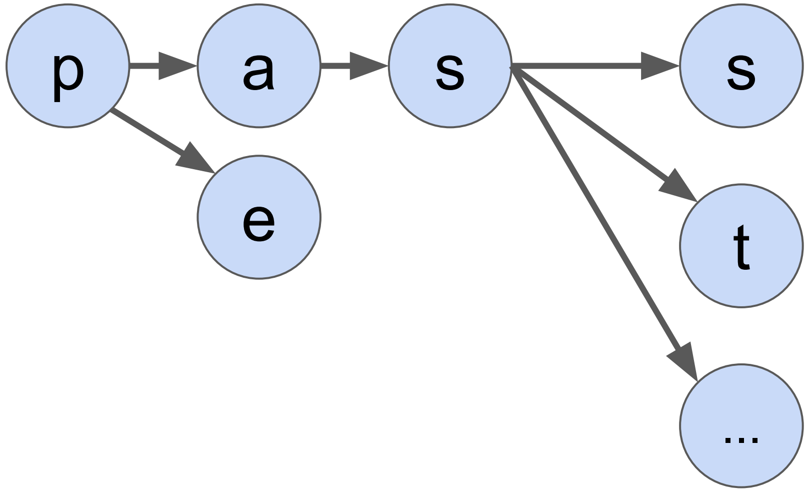
```
password11
password12
...
```

Guessing Methods

- John the Ripper
- Hashcat
- **Markov Models**

Guessing Methods

- John the Ripper
- Hashcat
- **Markov Models**

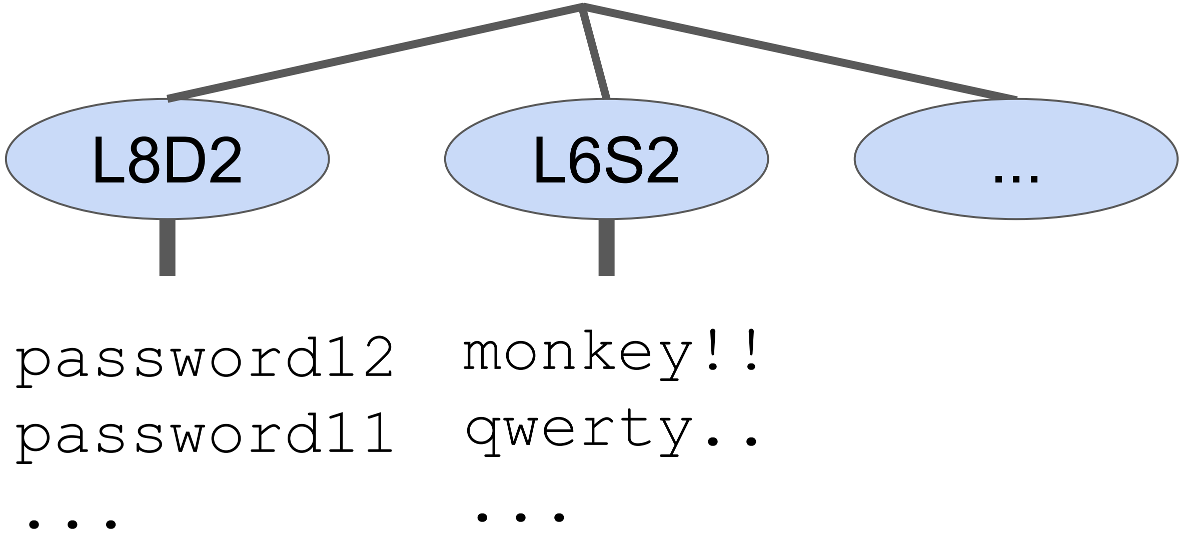


Guessing Methods

- John the Ripper
- Hashcat
- Markov Models
- **PCFGs**

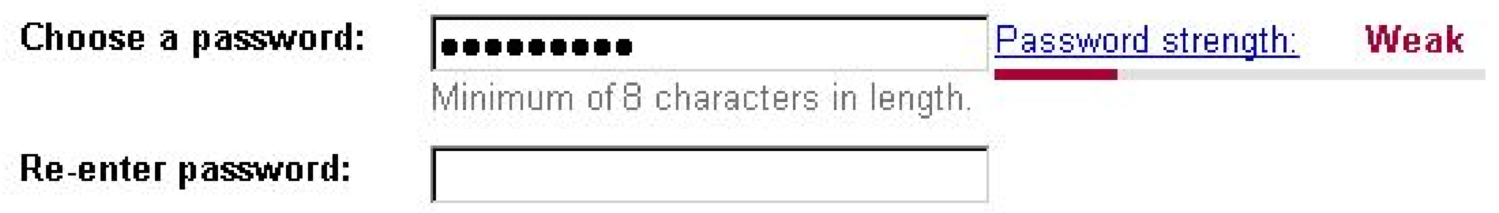
Guessing Methods

- John the Ripper
- Hashcat
- Markov Models
- **PCFGs**



Guessing Methods

- John the Ripper
- Hashcat
- Markov Models
- PCFGs



Choose a password: ●●●●●●●●●●

Minimum of 8 characters in length.

Password strength: **Weak**

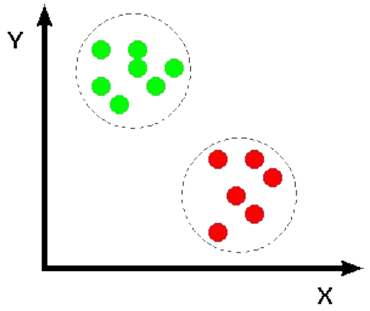
Re-enter password:

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` $\longrightarrow$ o or maybe 0 or O or ...

Generating Passwords

`passw` $\longrightarrow$

Next char is:

A: 3%

B: 1%

C: 0.6%

…

O: 55%

…

Z: 0.01%

0: 20%

1: ...

Generating Passwords

“”

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%

$\longrightarrow$

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:

A:	3%
B:	2%
[✓ C:]	[✓ 5%]
…	
O:	2%
…	
Z:	0.2%
0:	1%
1:	…
END:	2%

Generating Passwords

“C” $\longrightarrow$

Prob: 5%

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

"C"

Prob: 5%

$\longrightarrow$

Next char is:

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

Generating Passwords

“CA”
Prob: 0.5%

$\longrightarrow$

Next char is:

A:	3%
B:	10%
C:	7%
…	
O:	1%
…	
Z:	0.03%
0:	2%
1:	…
END:	12%

Generating Passwords

“CAB”
Prob: 0.05%

$\longrightarrow$

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%

$\longrightarrow$

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%

$\longrightarrow$

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

~~CAB - 0.006%~~

~~CAC - 0.0042%~~

ADD1 - 0.002%

CODE - 0.0013%

...

MUST BE LONGER THAN
3 CHARACTERS

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

`12345678912345678 9`

`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

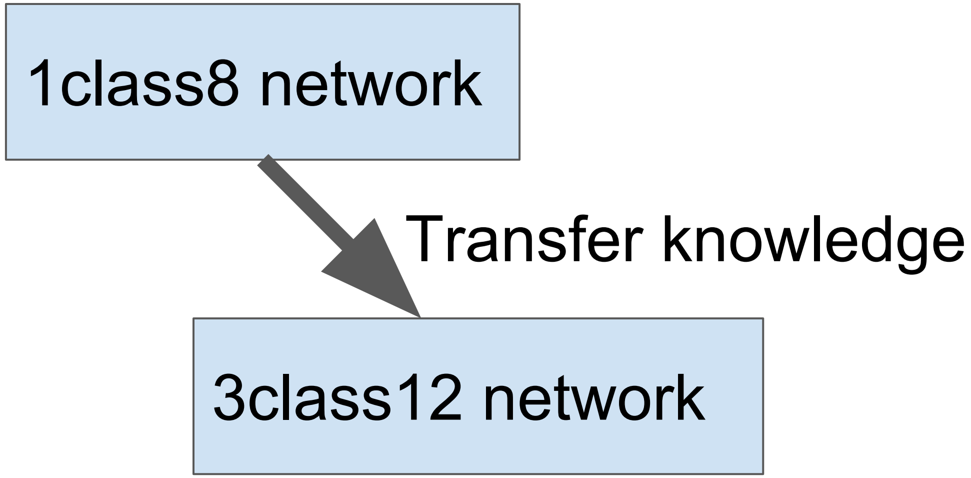
- **Model size**

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?

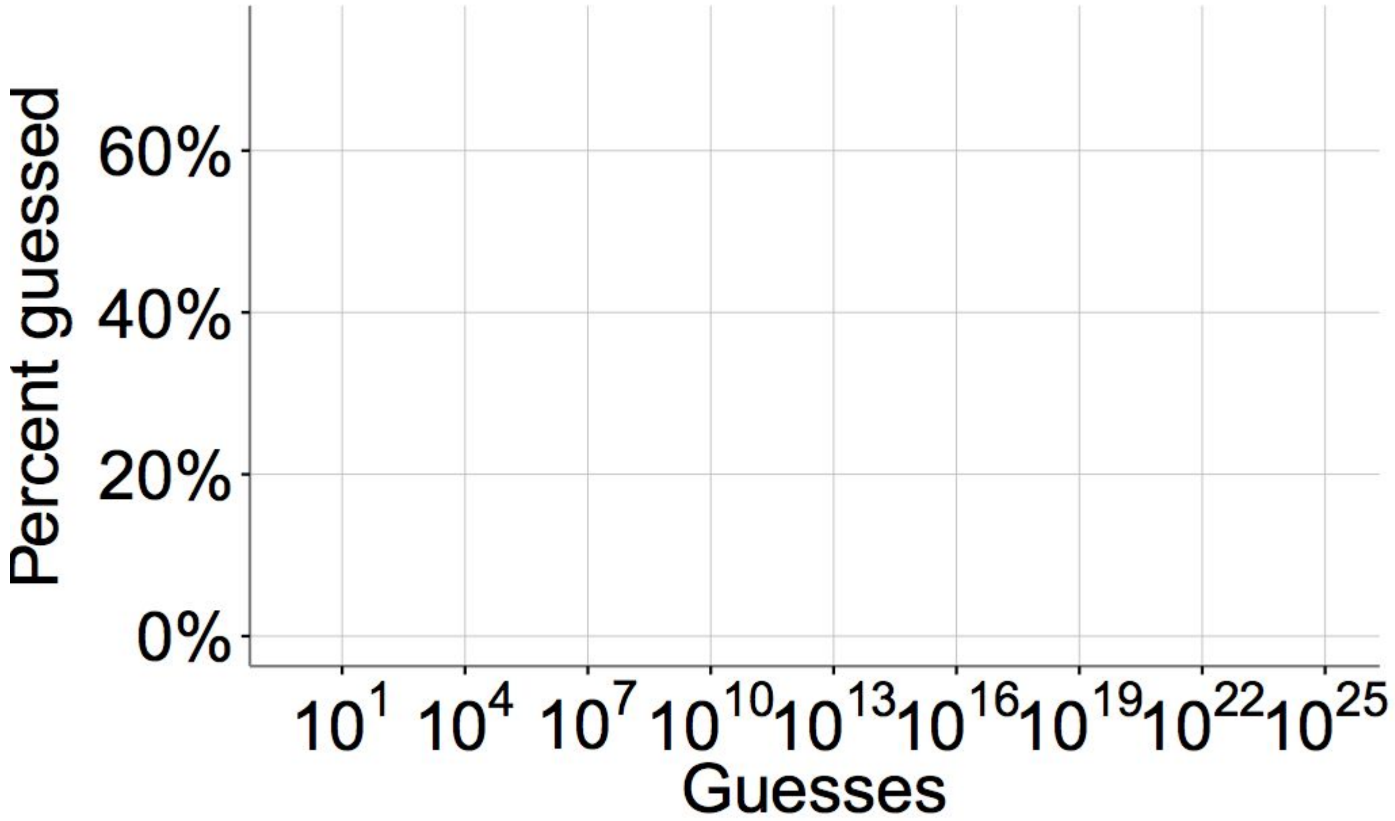
Design Space

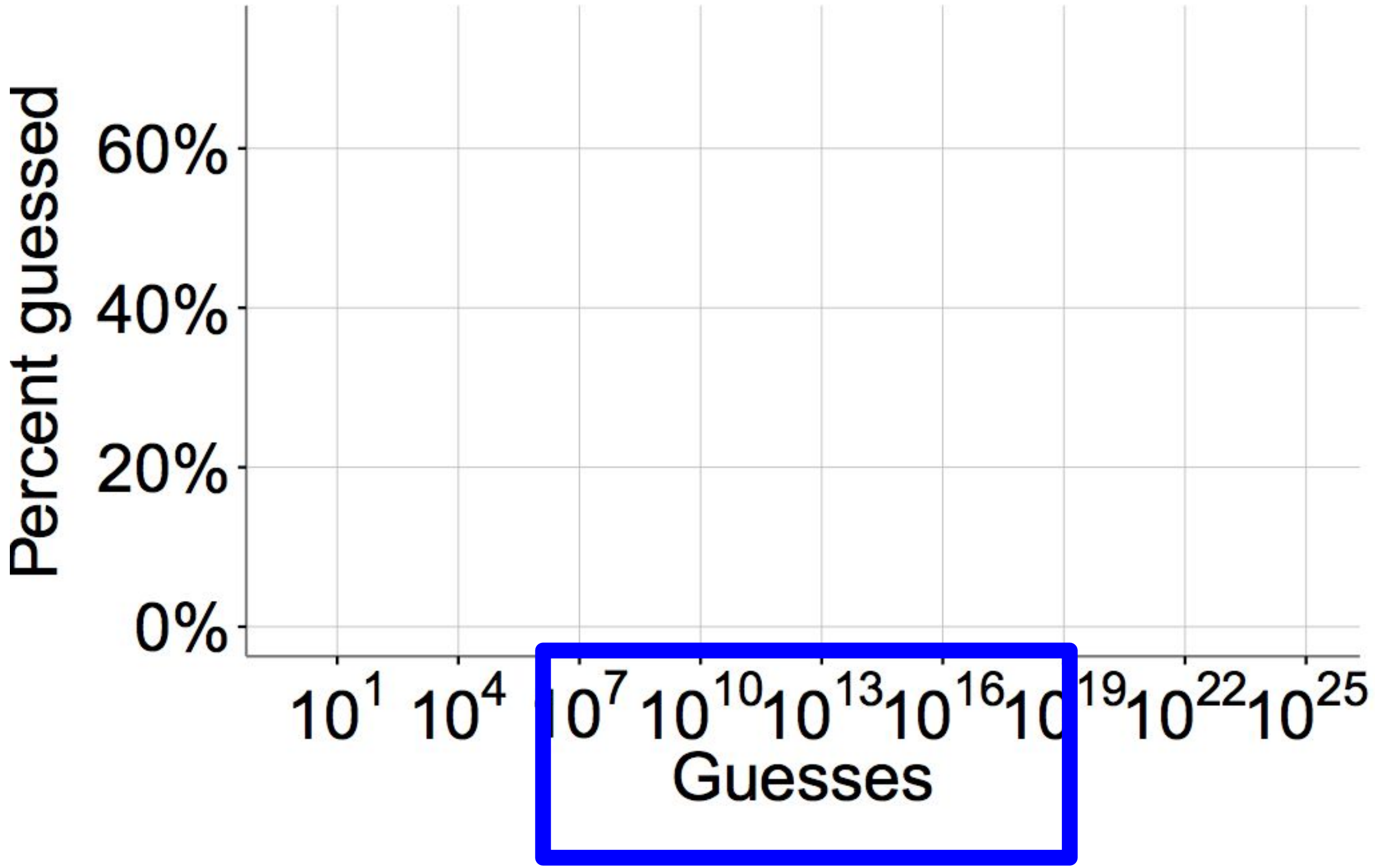
- Model size
- Transference learning
- Training data
- **Model architecture**
- **Alphabet size**
- **Password context**

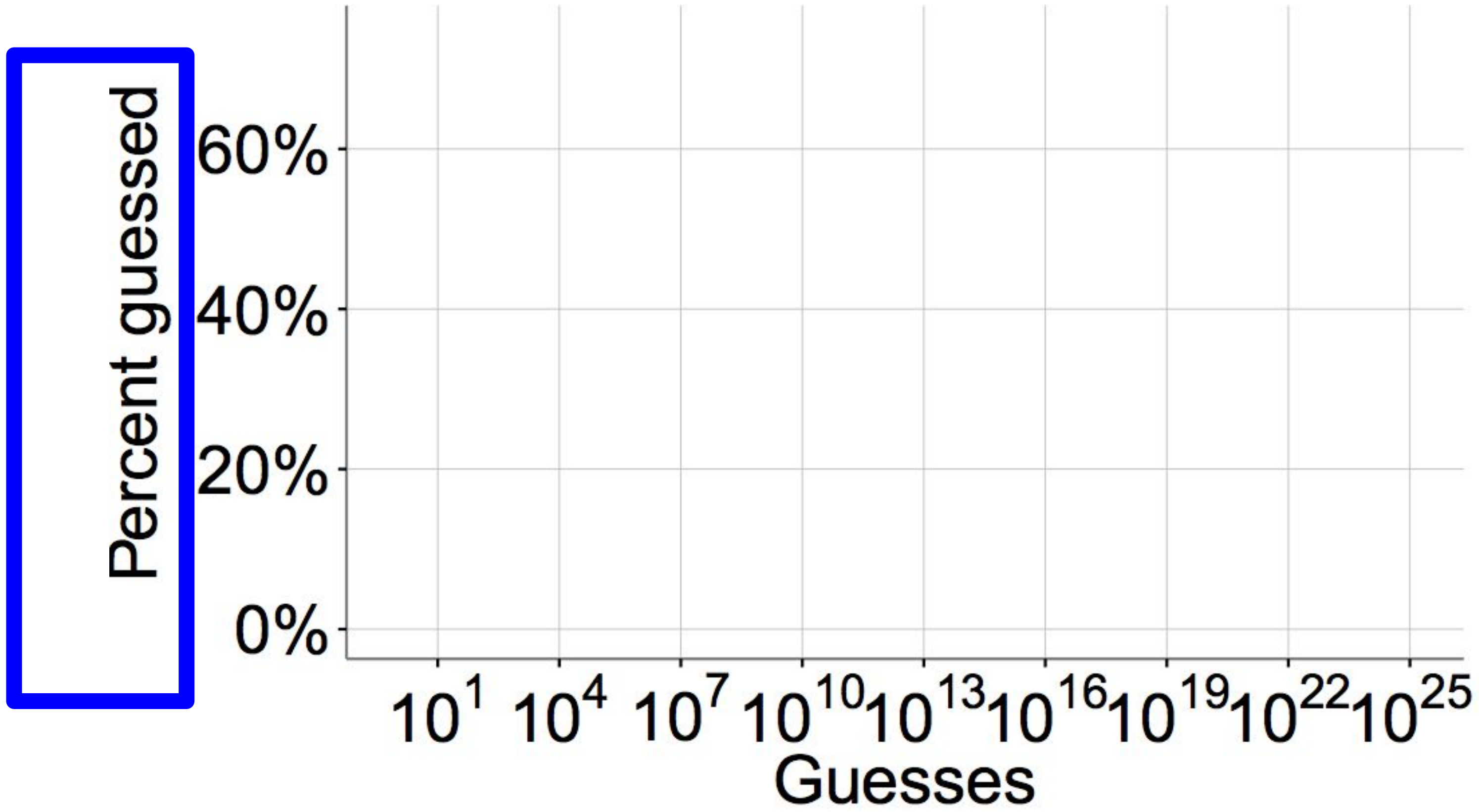
Testing Methodology

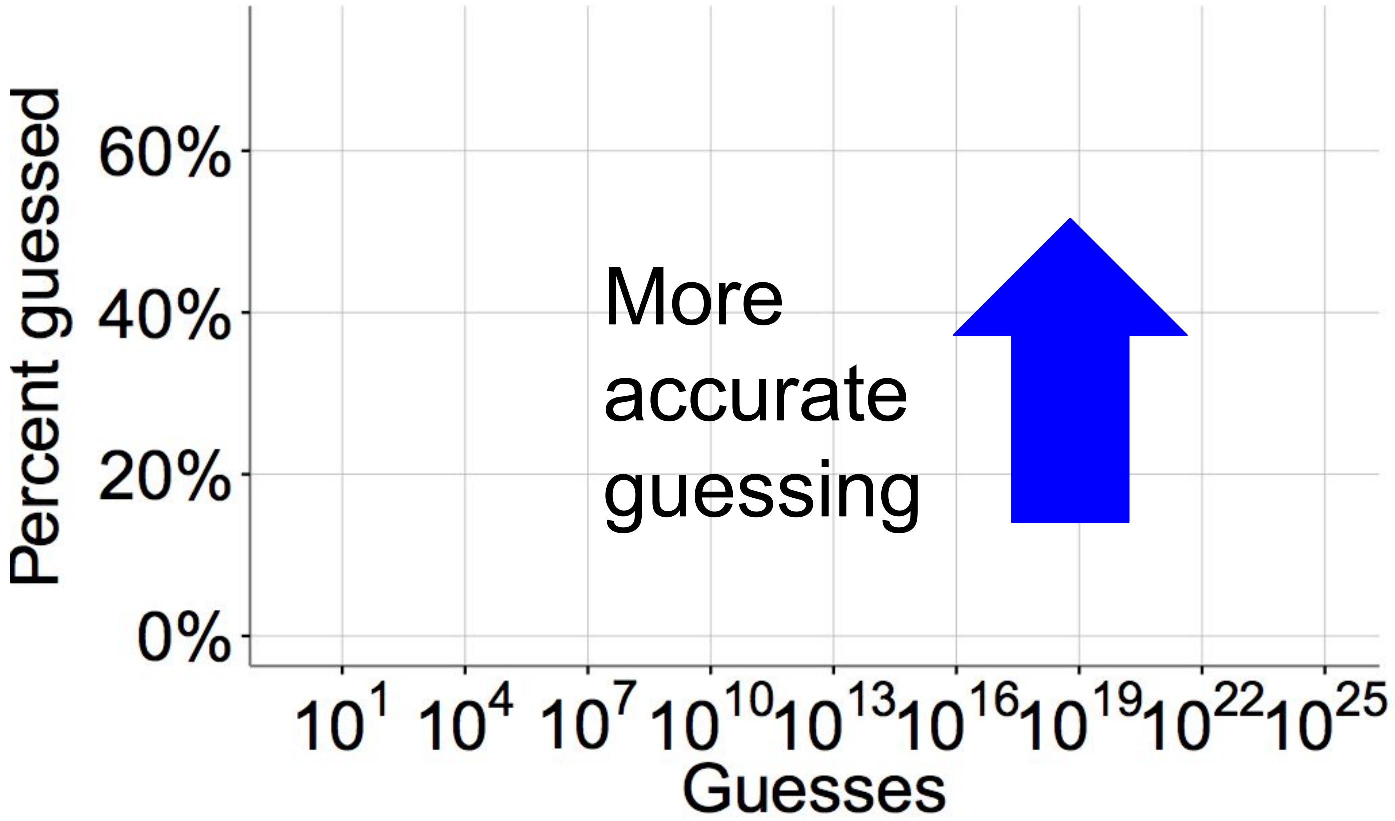
- Approach: measure # guessed passwords
- Training data: leaked password sets
- Testing data
    - MTurk study passwords: 1class8, 4class8, 1class16, 3class12
    - Real passwords: 000webhost password leak
- Use Monte-Carlo to estimate guess numbers (Dell’Amico and Filippone CCS ‘15)

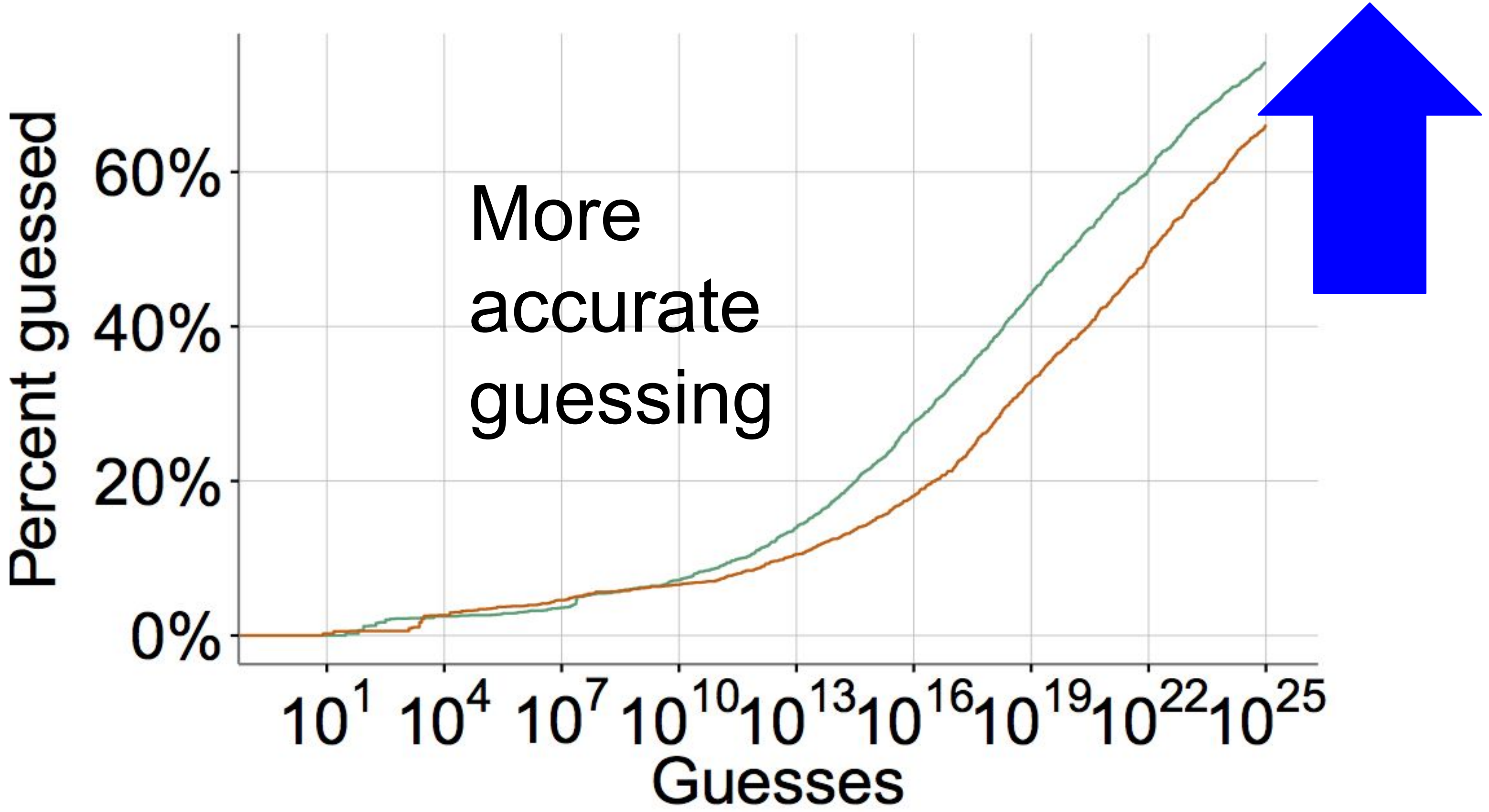
Tuning Training


Percent guessed
60%
40%
20%
0%
10^1 10^4 10^7 10^{10} 10^{13} 10^{16} 10^{19} 10^{22} 10^{25}
Guesses

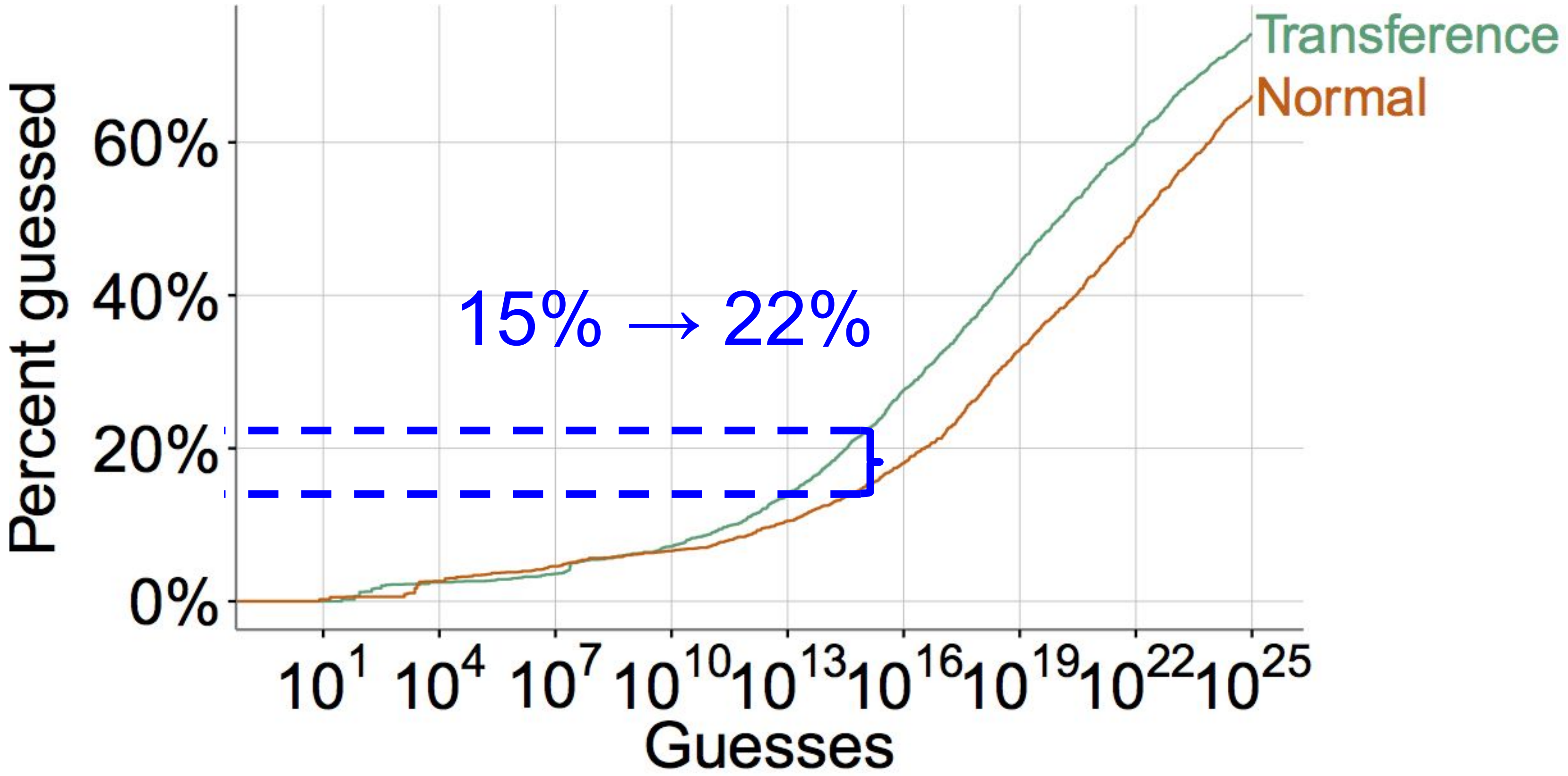

Percent guessed
60%
40%
20%
0%
10^1
10^4
10^7
10^{10}
10^{13}
10^{16}
10^{19}
10^{22}
10^{25}
Guesses


Percent guessed
60%
40%
20%
0%
10^1
10^4
10^7
10^{10}
10^{13}
10^{16}
10^{19}
10^{22}
10^{25}
Guesses

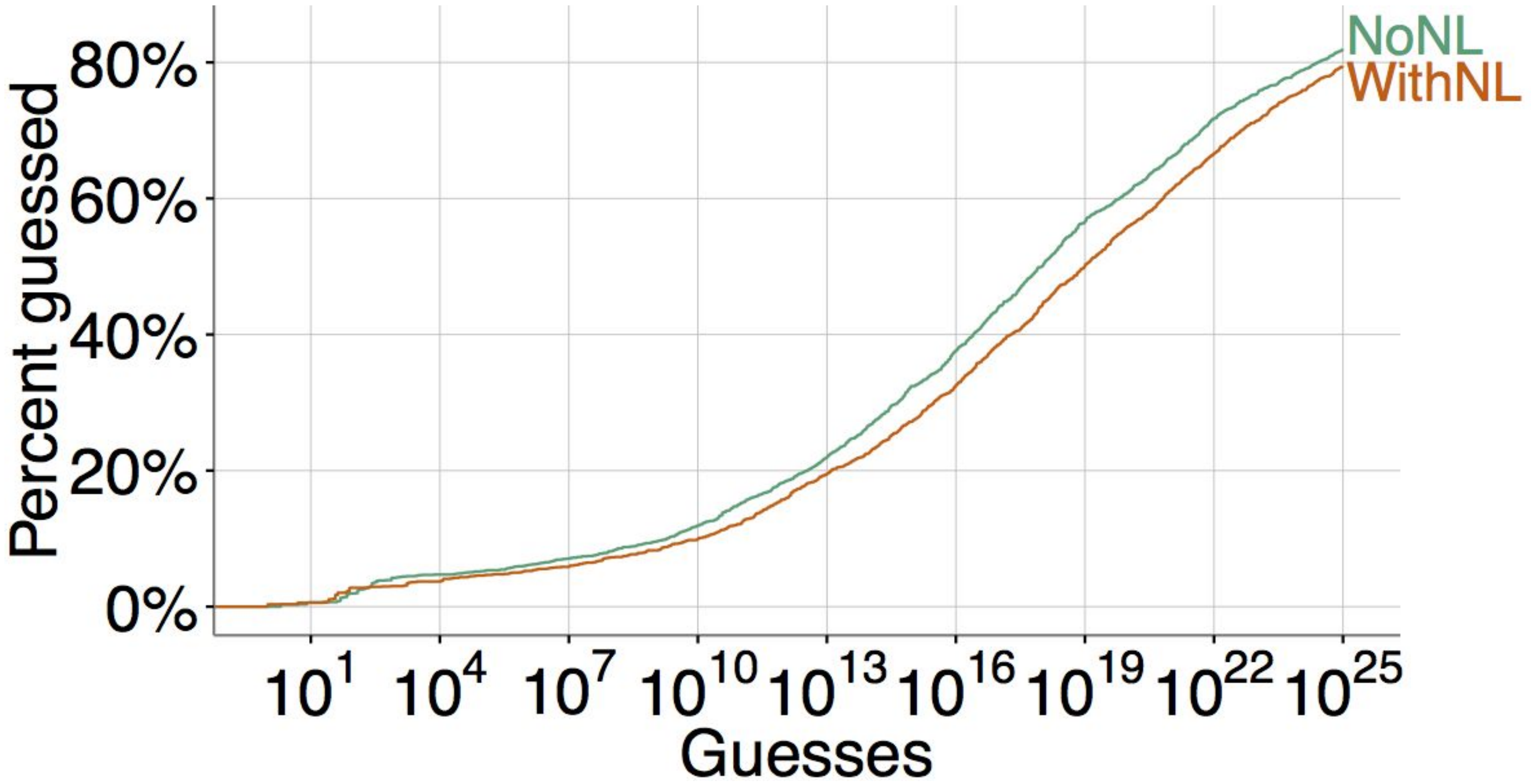

Percent guessed
60%
40%
20%
0%
10^1 10^4 10^7 10^10 10^13 10^16 10^19 10^22 10^25
Guesses
More accurate guessing


Percent guessed
60%
40%
20%
0%
10^1
10^4
10^7
10^{10}
10^{13}
10^{16}
10^{19}
10^{22}
10^{25}
Guesses
More accurate guessing

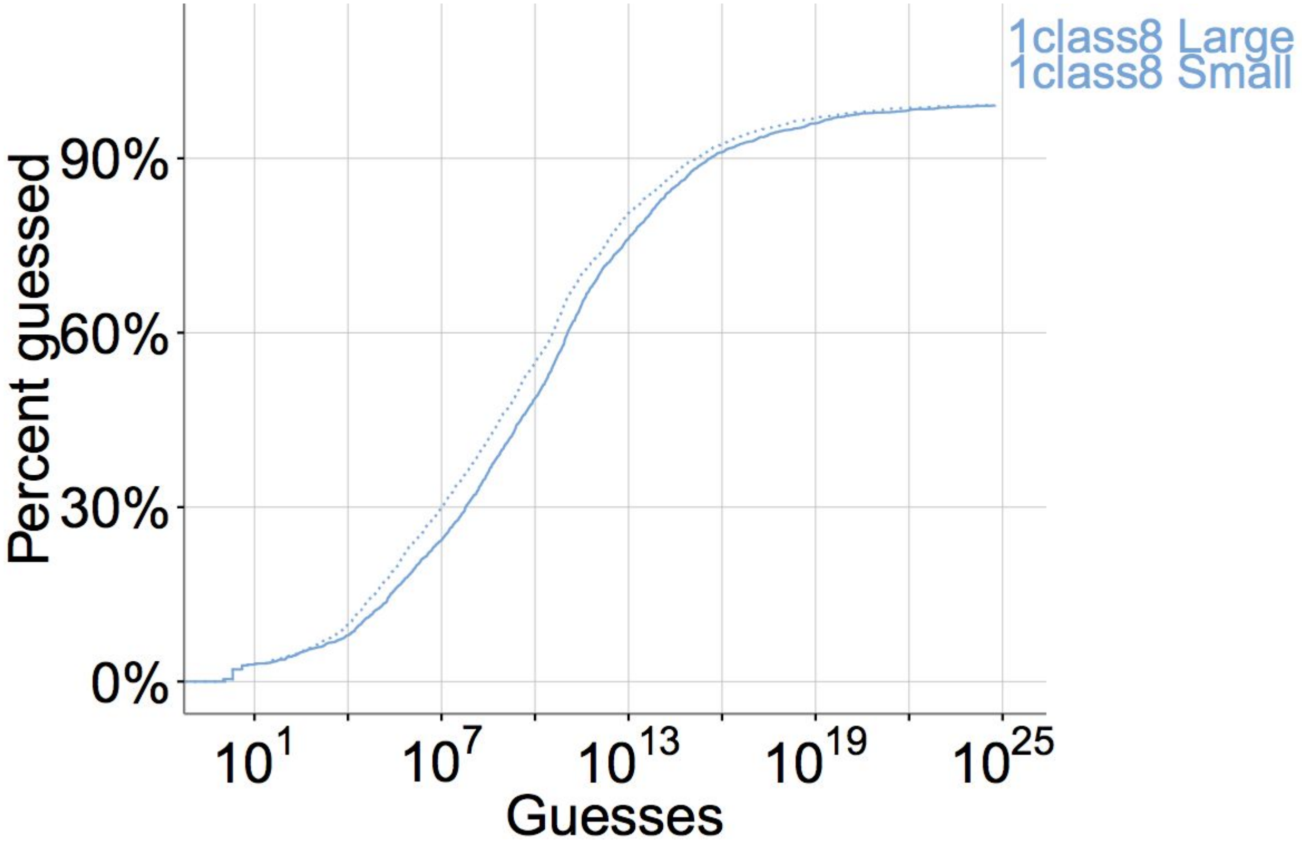
Transference Learning → More Accurate



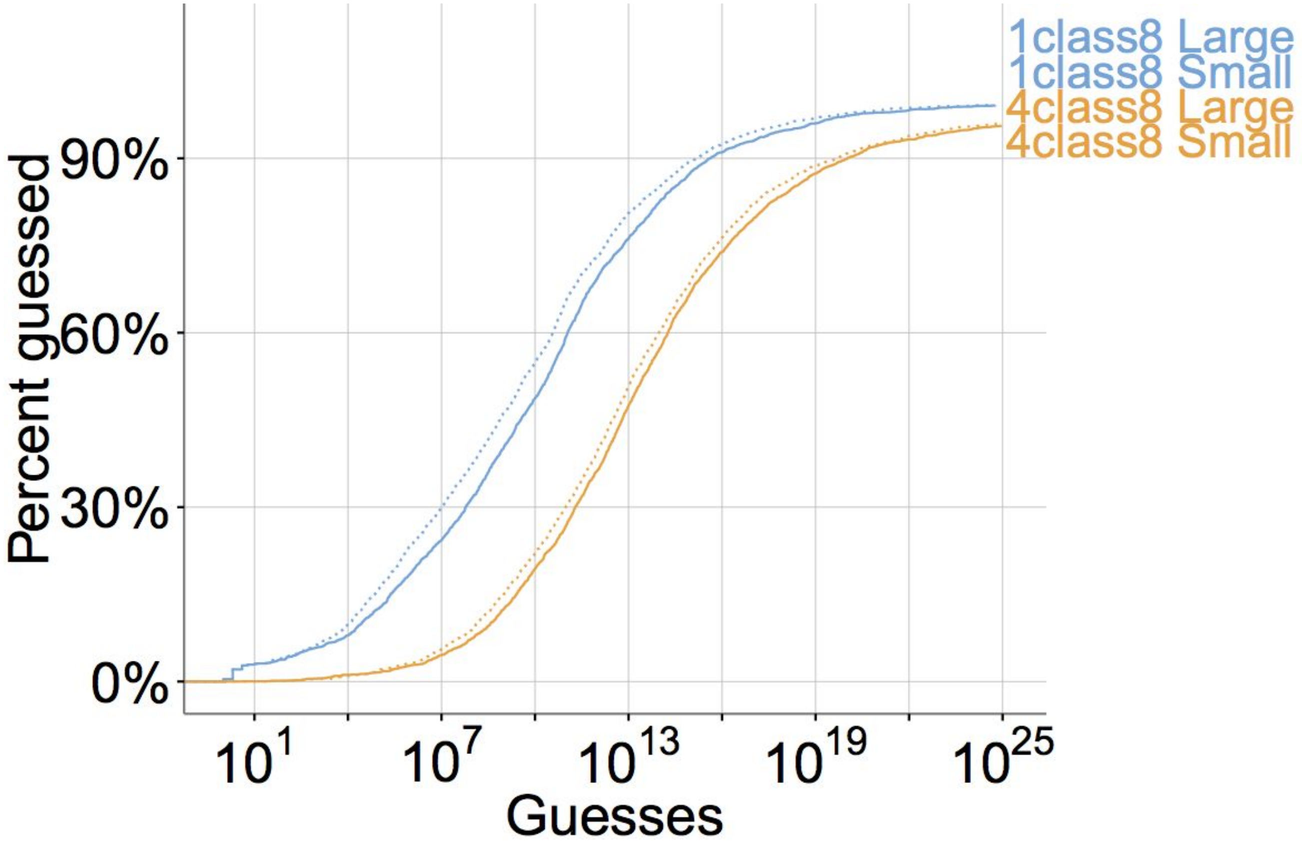
Natural Language Doesn’t Help



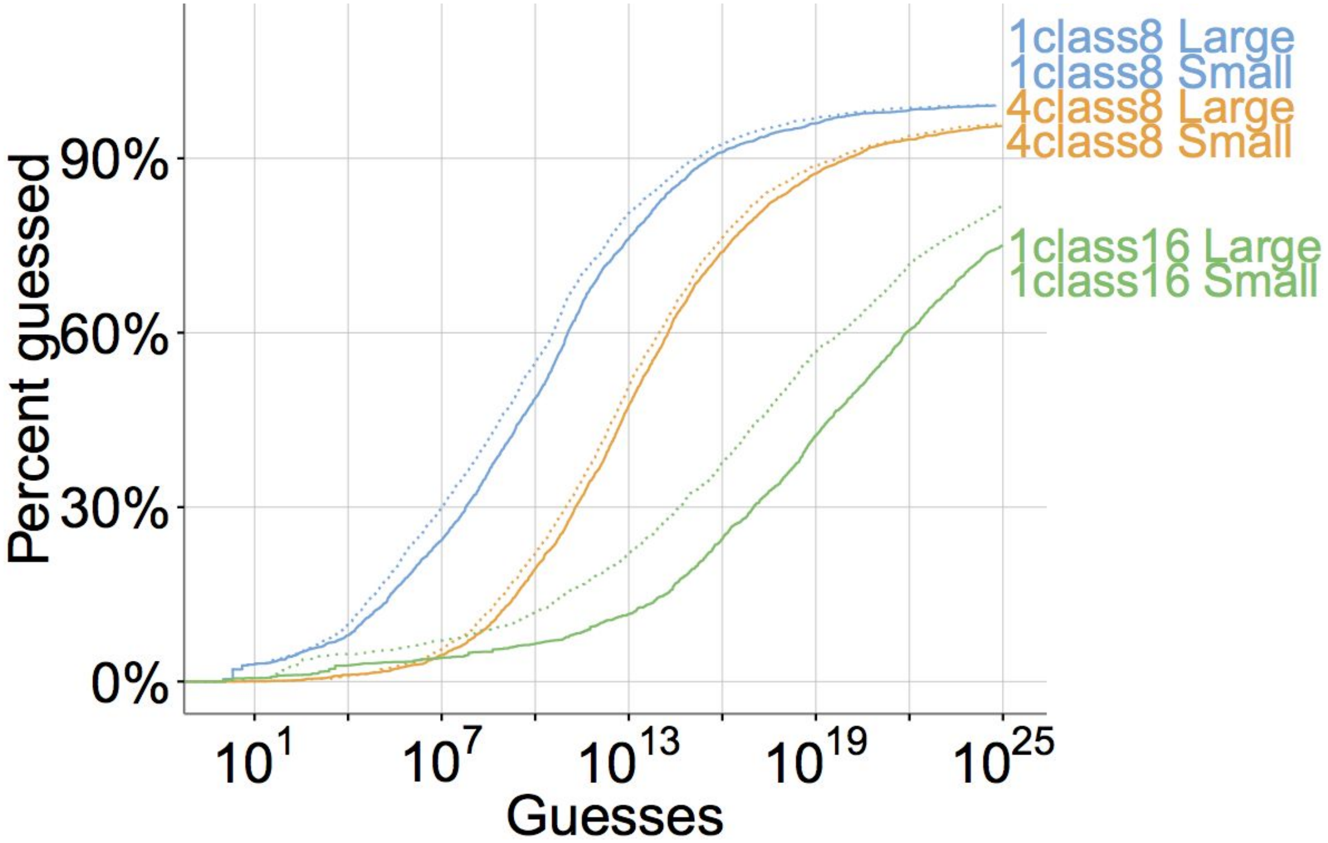
Model Size: Larger Is More Accurate



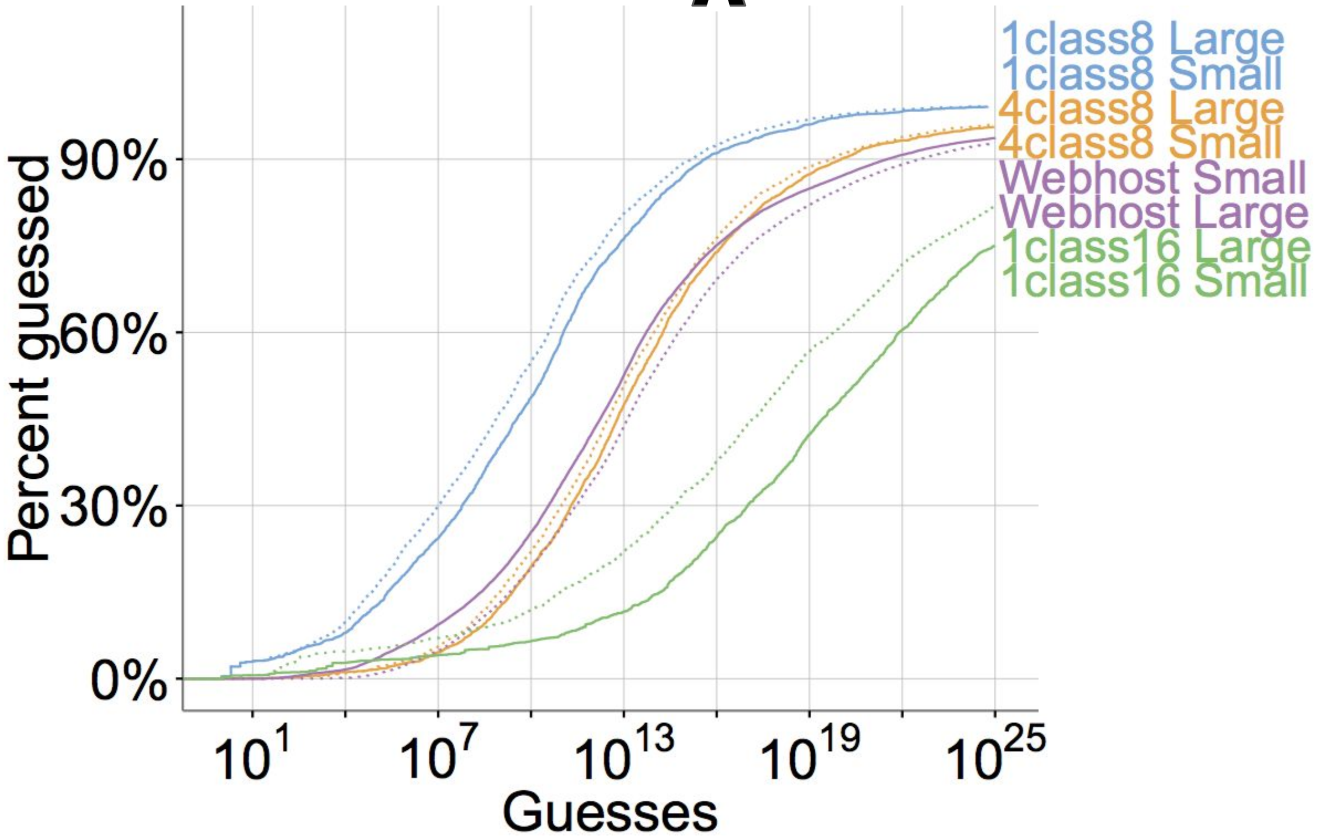
Model Size: Larger Is More Accurate



Model Size: Larger Is More Accurate

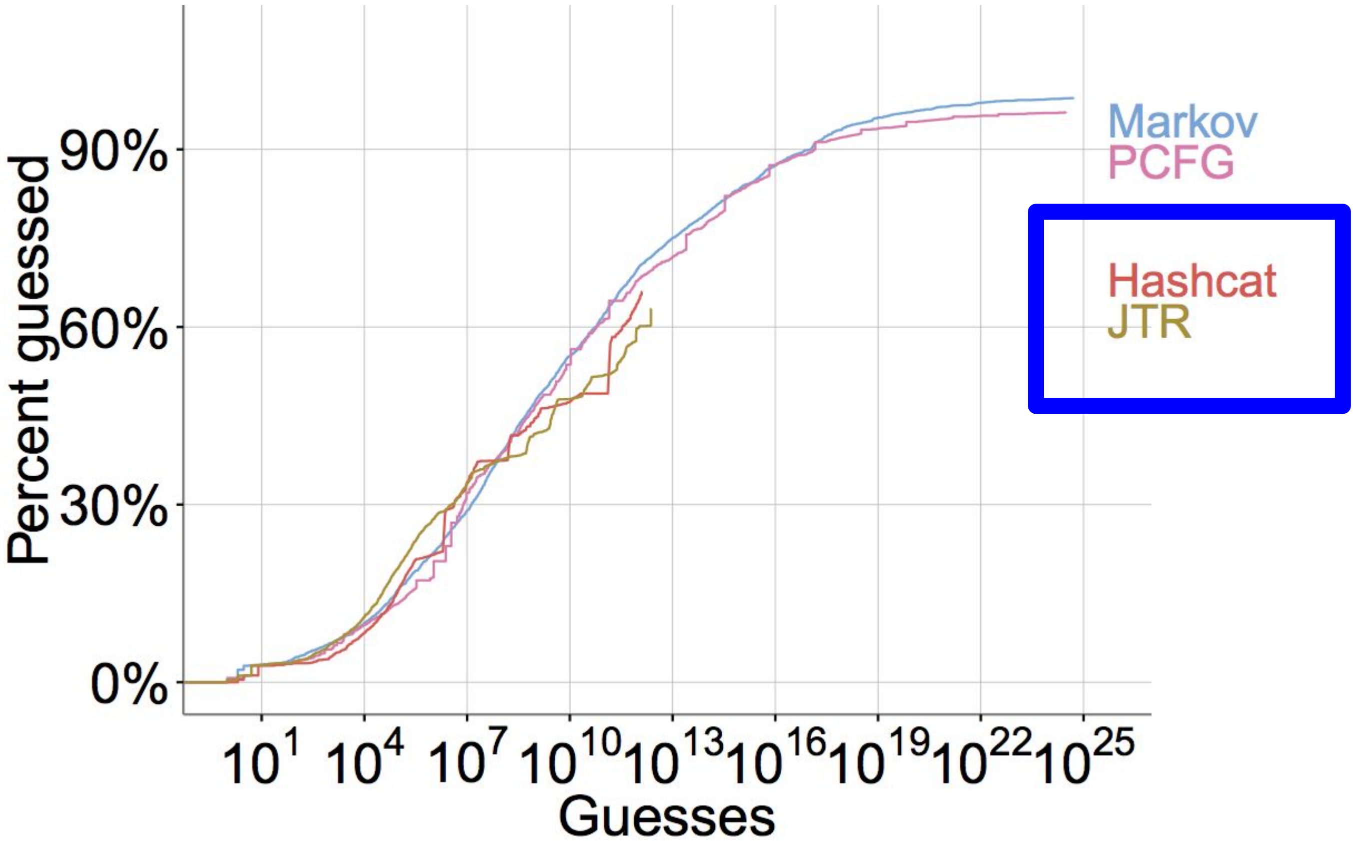


Model Size: Larger Is $\wedge$ Sometimes More Accurate

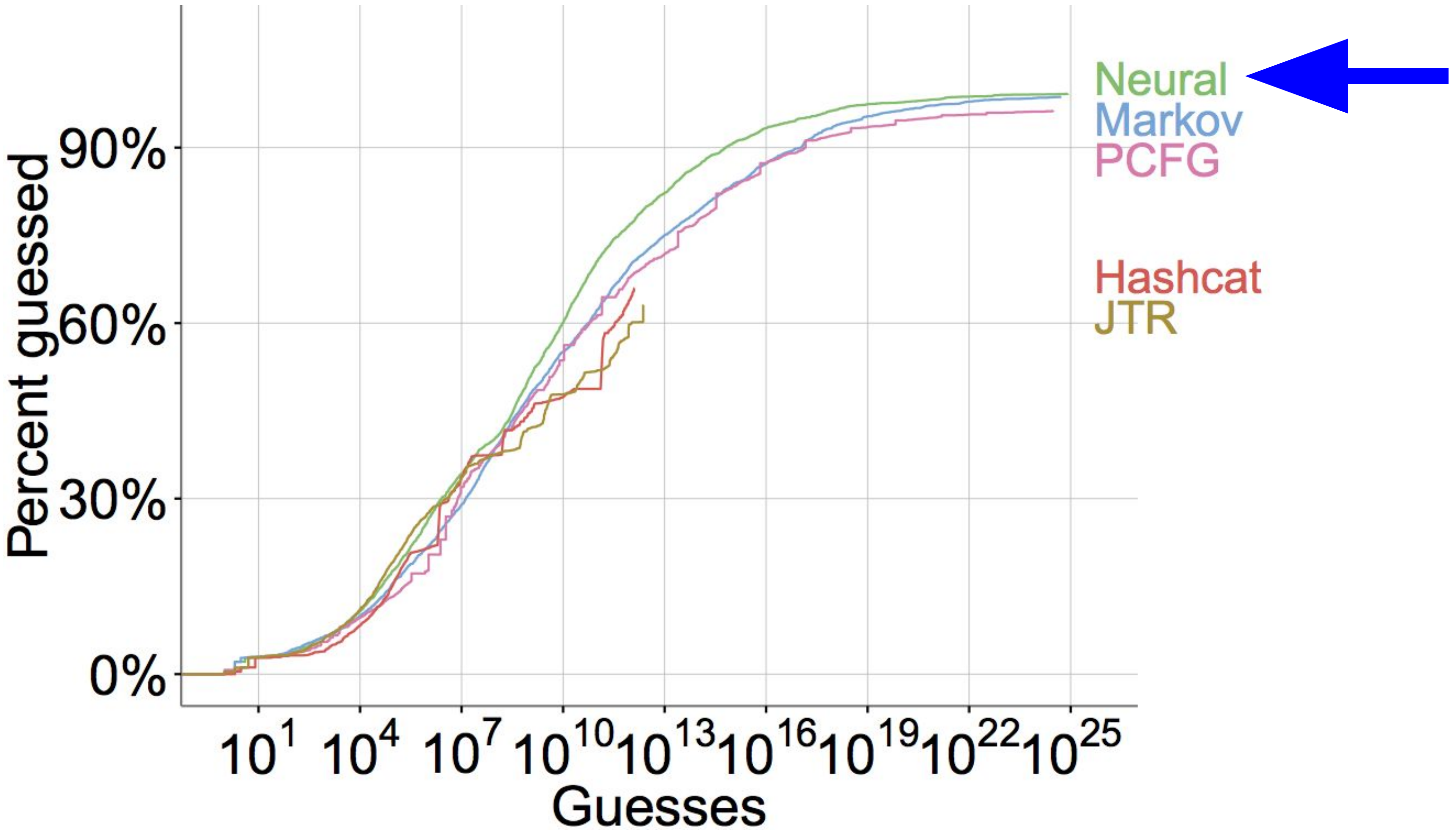


Comparison to Other Approaches

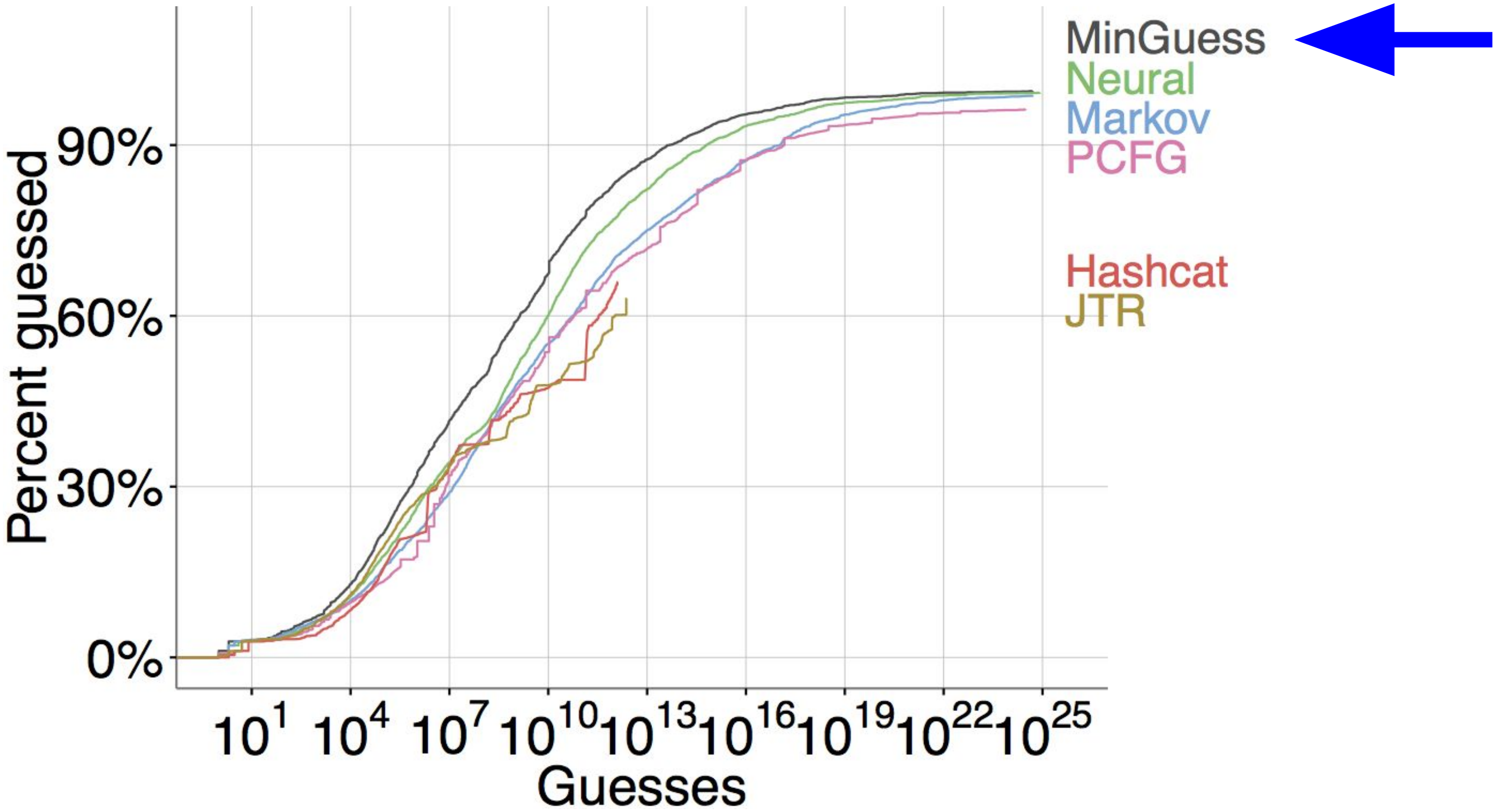
1class8: Comparison



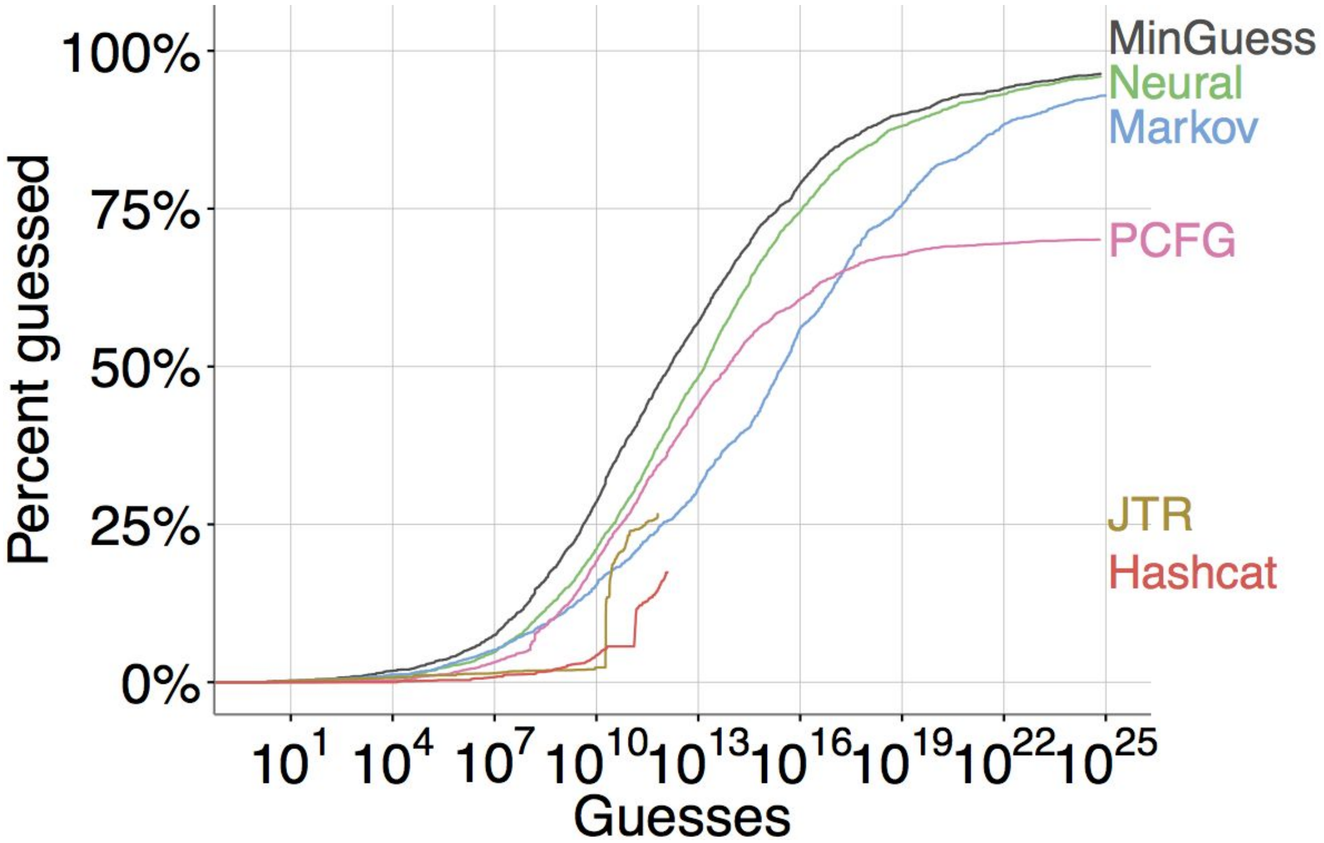
1class8: Neural Networks Guess Better



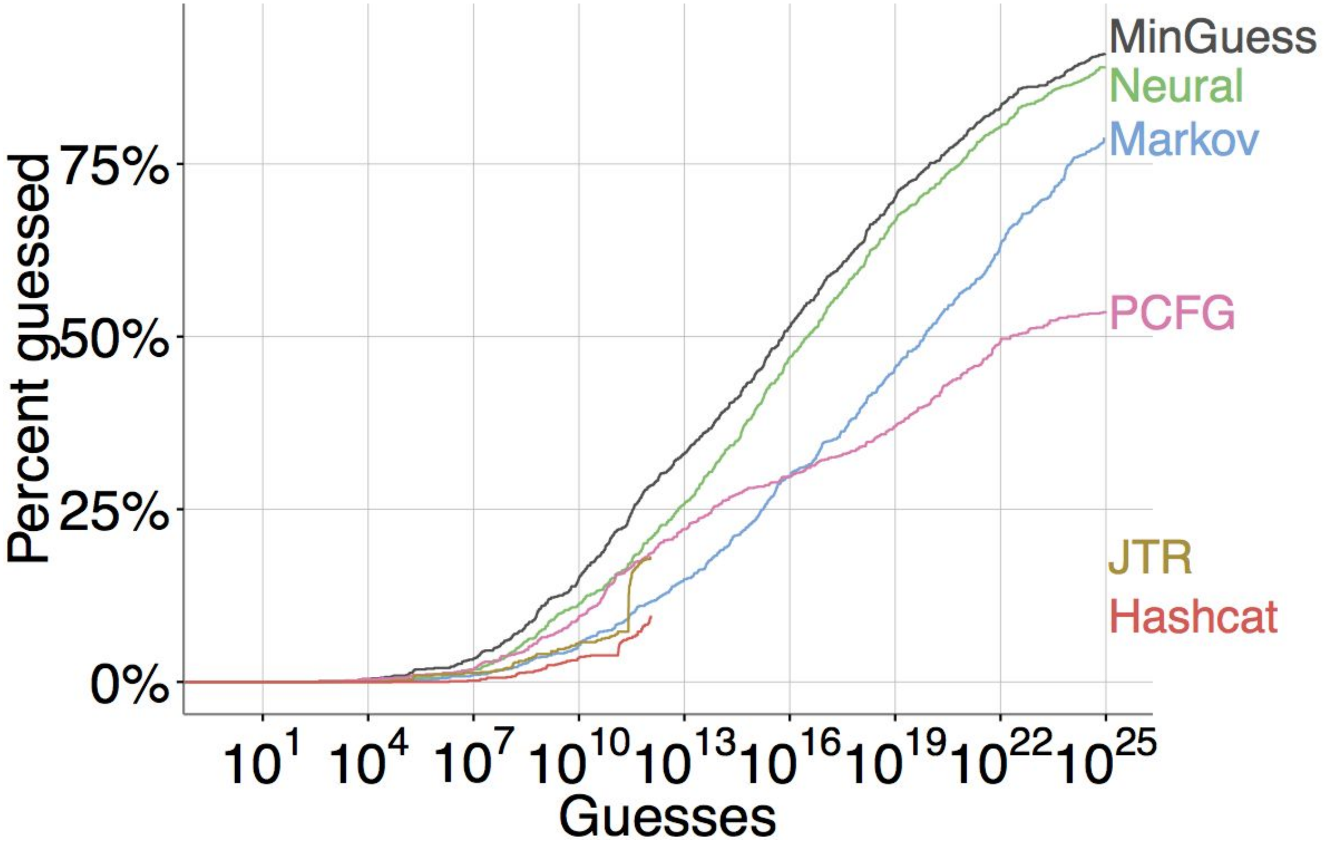
1class8: Neural Networks Guess Better



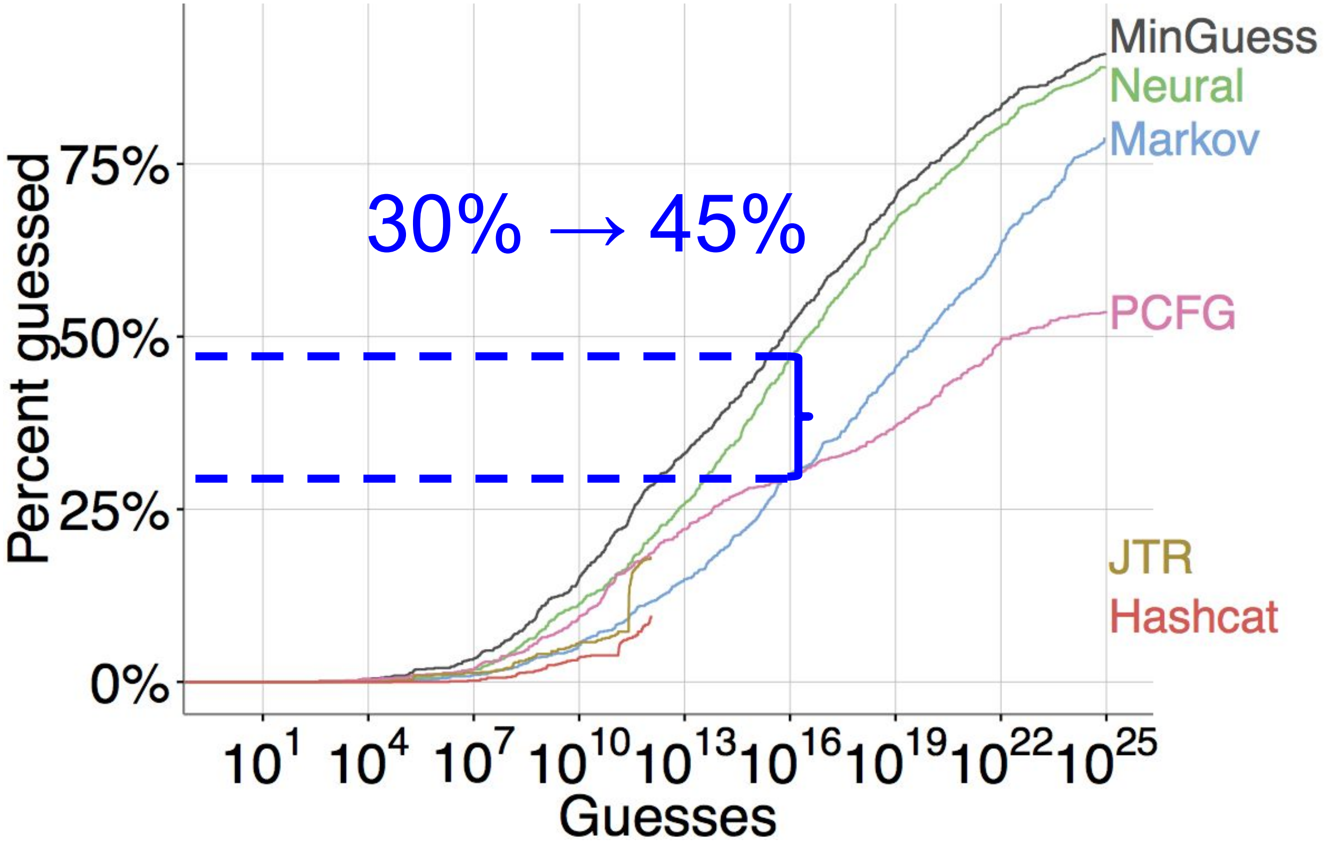
4class8: Neural Networks Guess Better



3class12: Neural Networks Guess Better



3class12: Neural Networks Guess Better



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

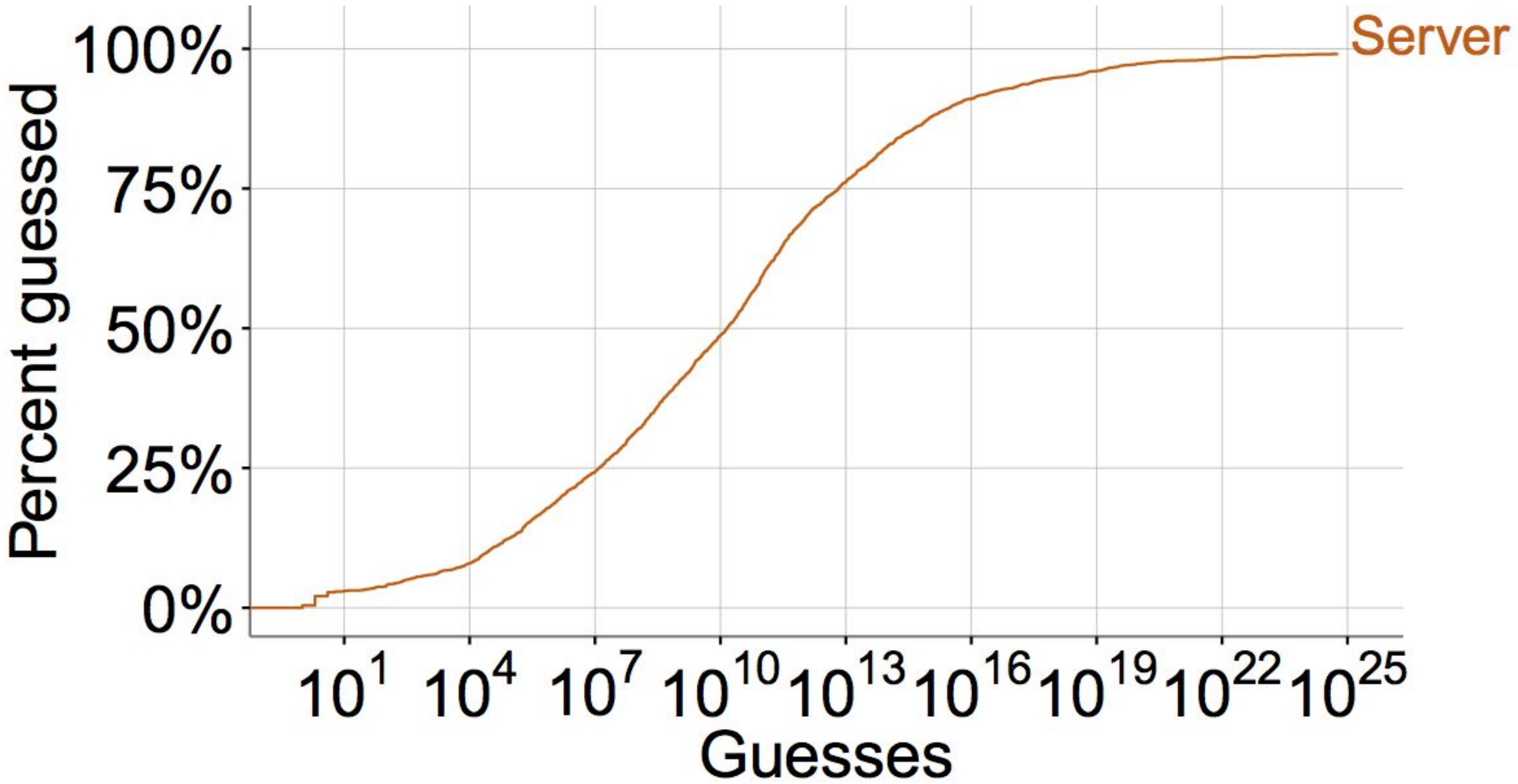
Making Meters Fast

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

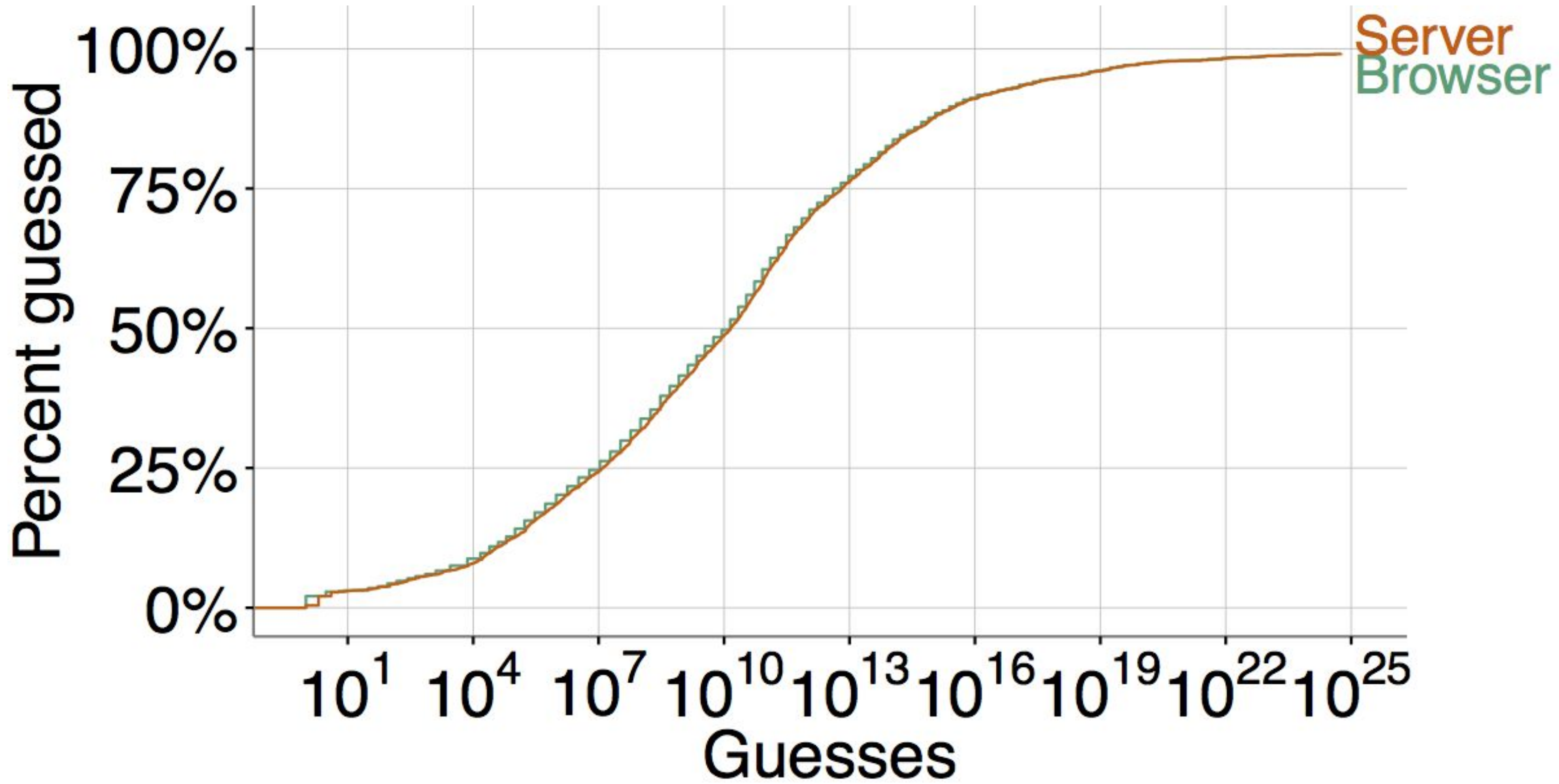
17 ms   < 0.1 sec

Meter Accuracy

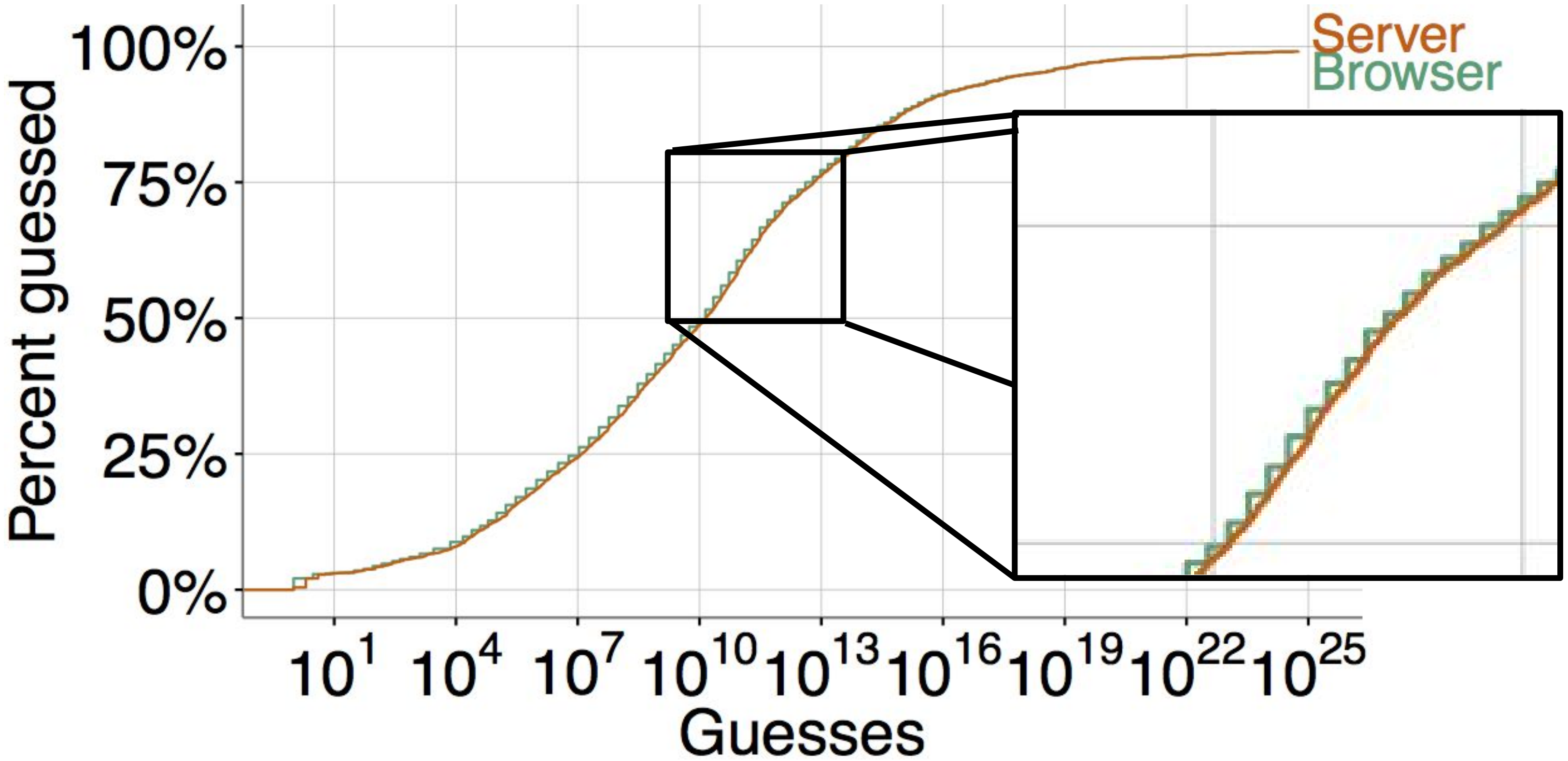
Meter Accuracy



Meter Accuracy



Meter Accuracy



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