Practical cache-based side channel attacks with JavaScript

Jack Cook
Last time...
Replicating the original paper

- Shusterman et al. found that many websites exhibit highly unique cache contention patterns.
- While a website loads (in a separate tab/window), we can repeatedly access values from memory and measure how long it takes to retrieve them.
- These can be visualized in a “memorygram”, where darker areas indicate more cache evictions over time.
- The uniqueness of these “memorygrams” can be exploited – a model trained on these can learn which website is being accessed.

Figure 3: Examples of memorygrams. Time progresses from left to right, shade indicates the number of evictions. (Darker shades correspond to more eviction.)
Most success with random forest models, which can be translated into JavaScript after they’re trained in Python
  - Makes the results a bit cooler – the website you’re on is displayed as soon as you open the page

If you want to detect the user opening between a small number of (around 4) websites, my work here is done
  - 100% accuracy when distinguishing between basically any set of 4 websites that I tried

At 10 different websites, accuracy drops to 90%

Still collecting data to distinguish between 100 different websites, which the original paper detected with 90% accuracy
The original paper made each trace 30 seconds long – I’ve found you can get almost all of the accuracy with about 2 seconds on a good Internet connection.
How it works
How to collect a website trace

- I found that trying to measure op/s gave better results than measuring cache contention
  - Also makes my code much easier to read
- Best results when this part was compiled to WASM
  - Interestingly, fewer op/s from WASM than JS, but results must have been more reliable
Demo (kind of)
TIL after losing her position in her university's anatomy department in 1938, Rita Levi-Montalcini set up a laboratory in her bedroom and studied the growth of nerve fibers in chicken embryos. This work led to her discovery of nerve growth factor, for which she was awarded a Nobel Prize in 1986.
Results
How long should traces be?
Results from this morning!

Accuracy is usually around 97% when classifying between the Alexa top 10.

When classifying between the Alexa top 50, accuracy drops to 74% (not amazing, but remember: our random choice baseline is 2%).

How many websites can we classify?

![Number of Websites vs. Accuracy](chart.png)
Can we predict traces on new computers?

- Up until now, I’ve been collecting training and testing data on my own computer
  - This gives great results, but is not representative of how this attack would probably be pulled off in the real world
- I collected testing data on my roommate’s Dell XPS 13 once, and the results were discouraging
  - Can only speculate why this is -- got unlucky? Differences due to OS? CPU?
- What if I could get data from a bunch of the same type of computer?
- Can I collect data on one MacBook Pro, and make accurate predictions on another identical MacBook Pro?
I asked all my friends to collect data

<table>
<thead>
<tr>
<th>Name</th>
<th>Screen</th>
<th>Year</th>
<th>Processor</th>
<th>OS Ver</th>
<th>Chrome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allen</td>
<td>13</td>
<td>2020</td>
<td>2.0 GHz i5</td>
<td>Catalina</td>
<td>86</td>
</tr>
<tr>
<td>Angela</td>
<td>13</td>
<td>2017</td>
<td>2.3 GHz i5</td>
<td>High Sierra</td>
<td>87</td>
</tr>
<tr>
<td>Jennifer</td>
<td>13</td>
<td>2017</td>
<td>2.3 GHz i5</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Anna</td>
<td>13</td>
<td>2018</td>
<td>2.3 GHz i5</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Gwinnie</td>
<td>13</td>
<td>2017</td>
<td>2.3 GHz i5</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Katherine</td>
<td>13</td>
<td>2018</td>
<td>2.3 GHz i5</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Jamie</td>
<td>16</td>
<td>2019</td>
<td>2.3 GHz i9</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Hassan</td>
<td>16</td>
<td>2019</td>
<td>2.3 GHz i9</td>
<td>Big Sur</td>
<td>86</td>
</tr>
<tr>
<td>Julia</td>
<td>15</td>
<td>2016</td>
<td>2.6 GHz i7</td>
<td>Big Sur</td>
<td>87</td>
</tr>
<tr>
<td>Britney</td>
<td>16</td>
<td>2019</td>
<td>2.6 GHz i7</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Hannah</td>
<td>16</td>
<td>2019</td>
<td>2.6 GHz i7</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Kevin</td>
<td>15</td>
<td>2018</td>
<td>2.6 GHz i7</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Natalie</td>
<td>13</td>
<td>early 2015</td>
<td>2.7 GHz i5</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Eric</td>
<td>13</td>
<td>early 2015</td>
<td>2.7 GHz i5</td>
<td>High Sierra</td>
<td>87</td>
</tr>
<tr>
<td>Soomin</td>
<td>13</td>
<td>2018</td>
<td>2.7 GHz i7</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Ethan</td>
<td>15</td>
<td>2017</td>
<td>2.9 GHz i7</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>NYT</td>
<td>13</td>
<td>2017</td>
<td>3.1 GHz i5</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Claire</td>
<td>13</td>
<td>2017</td>
<td>3.1 GHz i5</td>
<td>Catalina</td>
<td>87</td>
</tr>
<tr>
<td>Dad</td>
<td>13</td>
<td>2017</td>
<td>3.5 GHz i7</td>
<td>Catalina</td>
<td>87</td>
</tr>
</tbody>
</table>

Snoopy Setup Instructions

1. Make sure you have npm installed
   - If nothing comes up when you type `npm` in Terminal, install it here: https://www.npmjs.com/get-npm
2. Install Selenium if you don’t already have it
   - Enter `pip install selenium` into your Terminal
   - Then, open Chrome and go to chrome://version to check your Chrome version
   - Download ChromeDriver according to your Chrome version from https://chromedriver.chromium.org/downloads
   - Move chromedriver into your path, e.g. `sudo mv ~/Downloads/chromedriver /usr/local/bin`
3. Clone the project
   - `git clone https://github.com/jackcook/snoopy`
4. `cd` into the cloned directory and install dependencies
   - `cd snoopy`
   - `npm install`
5. Start the server
   - `npm start`
6. Make sure you’re ready to collect data:
   1. Plug your laptop into its charger
   2. Disable your screen saver: In System Preferences, go to Desktop & Screen Saver and select “Start after awakening”
<table>
<thead>
<tr>
<th>Name</th>
<th>96%</th>
<th>97%</th>
<th>12%</th>
<th>18%</th>
<th>22%</th>
<th>23%</th>
<th>27%</th>
<th>31%</th>
<th>32%</th>
<th>33%</th>
<th>34%</th>
<th>35%</th>
<th>38%</th>
<th>43%</th>
<th>45%</th>
<th>49%</th>
<th>61%</th>
<th>78%</th>
<th>89%</th>
<th>91%</th>
<th>100%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Angola</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jennifer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gaynelle</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anna</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Katherine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jamie</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hassan</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Julia</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Britney</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hannah</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kevin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Natalie</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eric</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Soomin</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ethan</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Claire</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dad</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>All</td>
<td>Male</td>
<td>Female</td>
<td>All</td>
<td>Male</td>
<td>Female</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-----</td>
<td>------</td>
<td>--------</td>
<td>-----</td>
<td>------</td>
<td>--------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Allen</td>
<td>66%</td>
<td>75%</td>
<td>75%</td>
<td>23%</td>
<td>23%</td>
<td>23%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Angela</td>
<td>25%</td>
<td>25%</td>
<td>25%</td>
<td>45%</td>
<td>45%</td>
<td>45%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jennifer</td>
<td>19%</td>
<td>19%</td>
<td>19%</td>
<td>31%</td>
<td>31%</td>
<td>31%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gaynrie</td>
<td>22%</td>
<td>22%</td>
<td>22%</td>
<td>32%</td>
<td>32%</td>
<td>32%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Anna</td>
<td>19%</td>
<td>19%</td>
<td>19%</td>
<td>31%</td>
<td>31%</td>
<td>31%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Katherine</td>
<td>19%</td>
<td>19%</td>
<td>19%</td>
<td>31%</td>
<td>31%</td>
<td>31%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jamie</td>
<td>31%</td>
<td>31%</td>
<td>31%</td>
<td>31%</td>
<td>31%</td>
<td>31%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hassan</td>
<td>23%</td>
<td>23%</td>
<td>23%</td>
<td>31%</td>
<td>31%</td>
<td>31%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The table shows accuracy rates for various groups, with percentages for both male and female categories.
### Cross-Computer Accuracies

<table>
<thead>
<tr>
<th></th>
<th>Allen</th>
<th>Angela</th>
<th>Jennifer</th>
<th>Gaynelle</th>
<th>Anna</th>
<th>Katherine</th>
<th>Jamie</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average</td>
<td>66%</td>
<td>70%</td>
<td>78%</td>
<td>82%</td>
<td>85%</td>
<td>90%</td>
<td>90%</td>
</tr>
</tbody>
</table>

** Britney **
- 32%     
- 67%     
- 59%     
- 66%     
- 49%     
- 40%     
- 72%     
- 68%     
- 62%     
- 99%     
- 57%     
- 19%     
- 47%     
- 44%     
- 52%     
- 67%     
- 43%     
- 20%

** Hannah **
- 48%     
- 25%     
- 23%     
- 23%     
- 30%     
- 27%     
- 57%     
- 49%     
- 43%     
- 73%     
- 98%     
- 30%     
- 24%     
- 21%     
- 48%     
- 81%     
- 28%     
- 20%

### Additional Names

- Natalie
- Eric
- Simon
- Ethan
- Claire
- Dad
### Cross-Computer Accuracies

<table>
<thead>
<tr>
<th>Name</th>
<th>15%</th>
<th>20%</th>
<th>25%</th>
<th>30%</th>
<th>35%</th>
<th>40%</th>
<th>45%</th>
<th>50%</th>
<th>55%</th>
<th>60%</th>
<th>65%</th>
<th>70%</th>
<th>75%</th>
<th>80%</th>
<th>85%</th>
<th>90%</th>
<th>95%</th>
<th>100%</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>95%</td>
<td>90%</td>
<td>85%</td>
<td>80%</td>
<td>75%</td>
<td>70%</td>
<td>65%</td>
<td>60%</td>
<td>55%</td>
<td>50%</td>
<td>45%</td>
<td>40%</td>
<td>35%</td>
<td>30%</td>
<td>25%</td>
<td>20%</td>
<td>15%</td>
<td>10%</td>
</tr>
<tr>
<td>Jane</td>
<td>95%</td>
<td>90%</td>
<td>85%</td>
<td>80%</td>
<td>75%</td>
<td>70%</td>
<td>65%</td>
<td>60%</td>
<td>55%</td>
<td>50%</td>
<td>45%</td>
<td>40%</td>
<td>35%</td>
<td>30%</td>
<td>25%</td>
<td>20%</td>
<td>15%</td>
<td>10%</td>
</tr>
<tr>
<td>Mary</td>
<td>95%</td>
<td>90%</td>
<td>85%</td>
<td>80%</td>
<td>75%</td>
<td>70%</td>
<td>65%</td>
<td>60%</td>
<td>55%</td>
<td>50%</td>
<td>45%</td>
<td>40%</td>
<td>35%</td>
<td>30%</td>
<td>25%</td>
<td>20%</td>
<td>15%</td>
<td>10%</td>
</tr>
<tr>
<td>Sue</td>
<td>95%</td>
<td>90%</td>
<td>85%</td>
<td>80%</td>
<td>75%</td>
<td>70%</td>
<td>65%</td>
<td>60%</td>
<td>55%</td>
<td>50%</td>
<td>45%</td>
<td>40%</td>
<td>35%</td>
<td>30%</td>
<td>25%</td>
<td>20%</td>
<td>15%</td>
<td>10%</td>
</tr>
<tr>
<td>Tim</td>
<td>95%</td>
<td>90%</td>
<td>85%</td>
<td>80%</td>
<td>75%</td>
<td>70%</td>
<td>65%</td>
<td>60%</td>
<td>55%</td>
<td>50%</td>
<td>45%</td>
<td>40%</td>
<td>35%</td>
<td>30%</td>
<td>25%</td>
<td>20%</td>
<td>15%</td>
<td>10%</td>
</tr>
</tbody>
</table>

---

**Note:** The table above shows the accuracy percentages for various users across different computer models. Each cell represents the accuracy percentage for a specific user-computer combination.
Predicting traces on new computers

- Even with training data from just one computer, I can get accuracies as high as 97% on other identical computers!

- Accuracy improved further when I combined data from multiple people with similar specs
  - Anna and Katherine have identical computers, but training on Anna’s data and testing on Katherine’s only gave 74% accuracy
  - When I trained on all 4 computers with 2.3 GHz i5 processors, accuracy jumped to 87%
Challenges

- Same problem as before: is poor prediction accuracy due to bad data, or a bad model?
- It’s hard to tell whether the way I’m making traces is the best one
  - My only way to tell if something improved is to collect data for many hours
  - This meant waking up, making a small adjustment to my trace collection, letting it run all day, making another adjustment in the evening, and then running it overnight while I was asleep... and then doing this for weeks
  - Probably wouldn't have been possible if I didn't have an old laptop with me
- Data is super noisy, and this is without any programs running in the background!
Progress!

- October 13: 87% accuracy between 4 websites
  - First working demo! Cache-based traces and a TensorFlow.js model
- October 28: 100% accuracy between 4 websites, 88% accuracy between 10 websites
  - Tweaked trace collection procedure, switched to random forest models
- November 10: 90% accuracy between 10 websites
  - Switched from cache-based traces to recording operations per second
- December 3: 94% accuracy between 10 websites
  - Fixed a bug with my selenium script, switched to extra trees classifier
- December 6: 97% accuracy between 10 websites
  - Compiled trace collection code to WebAssembly
- December 9: 74% accuracy between 50 websites
- Future: Can probably do better?
Future Work

- Investigate browsers other than Chrome
- Keep trying to find better ways to make traces
- Collect noisy data (e.g. while other applications are open) and see how much accuracy drops
- Distinguishing between websites opening and nothing happening at all (so that we don’t have to hit the start button to record a trace)
- Investigate differences due to network latency
- Make the 50-way classifier smaller...
project -- git-lfs - git push -- 80x23

```
jackcook@Jacks-MacBook-Pro project % git commit -m "Add updated classifier"
[master d73bfdc] Add updated classifier
  3 files changed, 12397584 insertions(+), 363791 deletions(-)
  rewrite classifier/memorygram.ipynb (91%)
jackcook@Jacks-MacBook-Pro project % git push
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (7/7), 27.58 MiB | 3.86 MiB/s, done.
  Total 7 (delta 3), reused 1 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
remote: error: Trace: 3685f1e524ba904f914fab2c64a1095b8022ce92be8a14b00a0cf11889c76b29
remote: error: See http://git.io/iEPT8g for more information.
remote: error: File classifier/classifier.js is 992.53 MB; this exceeds GitHub's file size limit of 100.00 MB
To github.com:jackcook/6-888-project.git
  ! [remote rejected] master -> master (pre-receive hook declined)
error: failed to push some refs to 'github.com:jackcook/6-888-project.git'
```
remote: **error:** File classifier/classifier.js is 992.53 MB; this exceeds GitHub's file size limit of 100.00 MB.

remote: **error:** Trace: 3685f1e528ab904f914fab2c641095b8022ce92be8a14b00a0cf11889c76b29
remote: **error:** See http://git.io/iEPt8g for more information.
remote: **error:** File classifier/classifier.js is 992.53 MB; this exceeds GitHub's file size limit of 100.00 MB.
To github.com:jackcook/6-888-project.git
  ! [remote rejected] master -> master (pre-receive hook declined)
error: failed to push some refs to 'git@github.com:jackcook/6-888-project.git'
Questions?