“Quantum” Classification of Malware

John Seymour (seymour1@umbc.edu)
2015-08-09
whoami

- Ph.D. student at the University of Maryland, Baltimore County (UMBC)
- Actively studying/researching infosec for about three years (mostly academic)
- Currently work for CyberPoint International
Outline

• D-Wave Basics
  • The D-Wave Controversy
  • How to play around on a D-Wave
• Some Machine Learning Background
• Building a “Quantum” Malware Classifier
QUANTUM COMPUTERS AND THE END OF SECURITY

SON

THOU HAST LET ME DOWN MIGHTILY
What you might have heard
(and why it’s wrong)

• **FALSE**: The D-Wave can solve NP-Complete problems in polynomial time.

• **PROBABLY FALSE**: The current D-Wave chip is already “better” than classical computing for hard problems.
The Current State of Affairs

Quantum effects are happening...
...but that might not be interesting

We don't know whether the D-Wave uses quantum effects for computation.

Regardless, it cannot run Shor's/Grover's/QKD.
D-Wave chips consist of:

• magnetized niobium loops
• couplers

http://www.dwavesys.com/
The D-Wave QUBO

\[ \sum_i a_i q_i + \sum_{i,j} b_{ij} q_i q_j \]

"Quadratic Unconstrained Binary Optimization"
They’ve got a website. To do stuff on and stuff.
System 6
One D-Wave Run

Input

Output
Blackbox/QSage

- Software, developed by D-Wave
- Turns arbitrary problems into QUBOs
- Heuristic-based (problem is NP-Complete)
- Conversation between classical machine and D-Wave
  - Adds time from network latency
print "creating remote connection"
url = 'https://gubist.dwavesys.com/sapi/'
token =
remote_connection = RemoteConnection(url, token)

print "starting BlackBox"
solver = remote_connection.get_solver('SYSTEM6')
obj = ObjectiveFunction()
blackbox_solver = BlackBoxSolver(solver)
blackbox_answer = blackbox_solver.solve(obj, num_vars)
blackbox_answer_bin = [int((item+1)/2) for item in blackbox_answer]
print "Best bitstring:",blackbox_answer_bin
So what can D-Wave machines do?

D-Wave claims applications: classification, protein-folding models, finding close-to-optimal solutions to NPC problems (e.g. Traveling Salesman)
Crash Course in Machine Learning
(At least what’s relevant to this)
Boosting - using combinations of weak classifiers

- Consider 3 classifiers with 70% accuracy

<table>
<thead>
<tr>
<th>All Correct:</th>
<th>Two Correct:</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.7 * 0.7 * 0.7 = 0.3429</td>
<td>3 * 0.7 * 0.7 * 0.3 = 0.4409</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Two Wrong:</th>
<th>All Wrong:</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 * 0.3 * 0.3 * 0.7 = 0.189</td>
<td>0.3 * 0.3 * 0.3 = 0.027</td>
</tr>
</tbody>
</table>

- Majority vote can increase your accuracy to 0.7838! (Hint: add up the top row)
- Most boosting algorithms also allow weights for these classifiers.

http://mlwave.com/kaggle-ensembling-guide/
Loss

- Want to minimize:
  - Number of misclassifications
  - Complexity of the model

\[
G(w) = \frac{1}{4} \sum_{s=1}^{S} (\text{sign} \left[ \sum_{j=1}^{D} w_j F_j(x_s) \right] - y_s)^2 + \lambda \sum_{j=1}^{D} w_j
\]
N-Grams

- Sliding window over text

DEADBEEF
N-Grams

- Sliding window over text

DEADBEEF
N-Grams

• Sliding window over text

DEADBEEF
N-Grams

- Sliding window over text

DEADBEEF
N-Grams

• Sliding window over text

  DEADBEEF

• Easy to generate
• Used before in malware with good results
• Easy to turn into weak classifiers
• Complex enough to compare classifiers
Building the “Quantum” Malware Classifier
QBoost

• Outperforms Adaboost
• Robust to label noise
  • Will generally still learn even if training data is mislabeled
  • Good for learning malware: ground truth is hard!
• Doesn’t scale to “Google-sized problems”
  • Blackbox supposedly
Dataset used

- Plenty of malicious datasets to choose from
  - Vx Heaven, VirusShare, scraping the web
  - We used Vx Heaven (fairly standard but old)
- No standard for benign dataset
  - Problematic
  - Windows + Cygwin + Sourceforge
    - Don’t do this
    - No adware was used in the making of this classifier
(Classical) Preprocessing

- Resample corpus to be balanced
  - Side-effects: Less time to train, lose information

- Extract Features (3-gram bytes)

HTTP://XKCD.COM/221/
At first, our classifier was no better than random chance

Next question: how long do we need to let Blackbox run?
• Previous work says 30 minute timeout

• Pilot Experiment to find how long it should take
  • Even on small problems, it takes 10 minutes to find decent solutions
  • Larger problems require even more time
• Limited to 32 features
  • We used 16 malware and 16 benign n-grams
  • Implemented QBoost with 10-fold cross-validation using Blackbox
    • Both on D-Wave and using a simulator
• Compared to several models from WEKA
  • Adaboost
  • J48 Decision Tree
  • Random Forest
## Results

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Cross-fold Accuracy</th>
<th>Average Time to Build (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-Wave</td>
<td>0.80</td>
<td>536.32</td>
</tr>
<tr>
<td>D-Wave Simulator</td>
<td>0.802</td>
<td>451.62</td>
</tr>
<tr>
<td>Adaboost</td>
<td>0.768</td>
<td>0.02</td>
</tr>
<tr>
<td>J48</td>
<td>0.796</td>
<td>0.03</td>
</tr>
<tr>
<td>RandomForest</td>
<td>0.814</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Table 1: Cross-fold accuracy and time to build classifiers.
## Results

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Cross-fold Accuracy</th>
<th>Average Time to Build (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-Wave</td>
<td>0.80</td>
<td>536.32</td>
</tr>
<tr>
<td>D-Wave Simulator</td>
<td>0.802</td>
<td>451.62</td>
</tr>
<tr>
<td>Adaboost</td>
<td>0.768</td>
<td>0.02</td>
</tr>
<tr>
<td>J48</td>
<td>0.796</td>
<td>0.03</td>
</tr>
<tr>
<td>RandomForest</td>
<td>0.814</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Table 1: Cross-fold accuracy and time to build classifiers.

**Interesting Result 1:** takes a LOT of time
Results

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Cross-fold Accuracy</th>
<th>Average Time to Build (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-Wave</td>
<td>0.80</td>
<td>536.32</td>
</tr>
<tr>
<td>D-Wave Simulator</td>
<td>0.802</td>
<td>451.62</td>
</tr>
<tr>
<td>Adaboost</td>
<td>0.768</td>
<td>0.02</td>
</tr>
<tr>
<td>J48</td>
<td>0.796</td>
<td>0.03</td>
</tr>
<tr>
<td>RandomForest</td>
<td>0.814</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Table 1: Cross-fold accuracy and time to build classifiers.

Interesting Result 2: Simulator > actual chip
What does it mean?

• Blackbox/D-Wave CAN learn a classifier
  • Probably a bad idea: significant overhead and must restrict problem substantially
  • We don’t know if this is due to Blackbox, the D-Wave chip, or some other issue
• General problem-solving on D-Wave is probably a bad idea
  • Really should’ve stuck with QBoost
• How much better is the next D-Wave chip?
• Embed directly onto chip, rather than use heuristics
• Better for another task?
  • e.g. feature preprocessing
• Machine Learning Standards for Malware Analysis
Thank you!

Questions?

seymour1@umbc.edu, @jjseymour3
References


Qloss: V. S. Denchev. Robust Classification with Adiabatic Quantum Optimization.
References
