Learning Noise in Quantum Information Processors

Travis L Scholten @Travis_Sch

Center for Quantum Information and Control, University of New Mexico, Albuquerque, USA

Center for Computing Research, Sandia National Laboratories, Albuquerque, USA

QTML 2017
There are lots of applications at the intersection of QI/QC and ML...
...I want to focus on how ML can improve characterization of quantum hardware.

Biamonte, et. al, arXiv: 1611.09347
Quantum device characterization (QCVV) techniques arranged by amount learned and time required.
Tomography is very informative, but time-consuming!

- Gate set tomography
- Process tomography
- State tomography

<table>
<thead>
<tr>
<th>Speed of learning</th>
<th>Amount learned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fast</td>
<td>Full</td>
</tr>
<tr>
<td></td>
<td>Limited</td>
</tr>
</tbody>
</table>
Randomized benchmarking is fast, but yields limited information.

<table>
<thead>
<tr>
<th>Speed of learning</th>
<th>Amount learned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slow</td>
<td>Limited</td>
</tr>
<tr>
<td>Fast</td>
<td>Full</td>
</tr>
</tbody>
</table>

- Gate set tomography
- Process tomography
- State tomography
- Randomized Benchmarking

(Several variants, leads to different kinds of information learned.)
Depending on how much we want to learn, and how quickly, machine learning could be useful.

Caveat: “Speed” doesn’t include *training time*
Depending on how much we want to learn, and how quickly, machine learning could be useful.

Can machine learning extract information about noise affecting near and medium-term quantum hardware?
Noise affects the outcome probabilities of quantum circuits.

How can we learn about noise using the data we get from running quantum circuits?
Noise in quantum hardware affects the outcome probabilities of circuits.

Example: over-rotation error of a single-qubit gate

\[ |0\rangle \xrightarrow{Y_{\pi/2}} \frac{\sin \theta}{2} \]

(The circuit we write down)

\[ \Pr(0) = \text{Tr}(|0\rangle\langle0|E(|0\rangle\langle0|)) = \frac{1}{2} (1 - \sin \theta) \]

(Noise affects outcome probability)
Gate set tomography (GST) provides a set of structured circuits we can use for learning.

GST assumes the device is a black box, described by a gate set.

GST prescribes certain circuits to run that collectively amplify all types of noise.

\[ |0\rangle \xrightarrow{Y_{\pi/2}} |\rangle \]

\[ |0\rangle \xrightarrow{Y_{\pi/2}} Y_{\pi/2} \xrightarrow{\text{}} \]

Standard use: Outcome probabilities are analyzed by pyGSTi software to estimate the noisy gates.

Blume-Kohout, et. al, arXiv 1605.07674
Gate set tomography (GST) provides a set of structured circuits we can use for learning.

GST prescribes certain circuits to run that collectively *amplify all types of noise*.

\[
\begin{align*}
|0\rangle & \xrightarrow{Y_{\pi/2}} l = 1 \\
|0\rangle & \xrightarrow{Y_{\pi/2}} Y_{\pi/2} l = 2 \\
|0\rangle & \xrightarrow{Y_{\pi/2}} Y_{\pi/2} Y_{\pi/2} Y_{\pi/2} l = 4
\end{align*}
\]

Circuits have varying length, up to some maximum length \( L \).

\[
l = 1, 2, 4, \cdots , L
\]

Why? Longer circuits are more sensitive to noise.
To do machine learning on GST data sets, embed them in a feature space.

\[ \mathbf{f} = (f_1, f_2, \cdots) \in \mathbb{R}^d \]

The dimension of the feature space grows with \( L \) because more circuits are added.
Noise changes some components of the feature vectors.

How can we identify the “signature” of a noise process using GST feature vectors?
**Principal component analysis** (PCA) is a useful tool for understanding the structure of GST feature vectors.

PCA finds a low-dimensional representation of data by looking for directions of *maximum variance*.

Compute covariance matrix & diagonalize

\[
C = \sum_{j=1}^{K} \sigma_j \sigma_j^T
\]

\[
\sigma_1 \geq \sigma_2 \cdots \geq \sigma_K
\]

Defines a map:

\[
f \rightarrow \sum_{j=1}^{K} (f \cdot \sigma_j) \sigma_j
\]
Projection onto a 2-dimensional PCA subspace reveals a structure to GST feature vectors.

Different noise types and noise strengths tend to **cluster**!

(PCA performed on entire dataset, then individual feature vectors transformed.)

<table>
<thead>
<tr>
<th>Noise Type</th>
<th>Number of Feature Vectors</th>
<th>Number of Noise Strengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coherent Error</td>
<td>450</td>
<td>9</td>
</tr>
<tr>
<td>Depolarization</td>
<td>450</td>
<td>9</td>
</tr>
<tr>
<td>No Noise</td>
<td>50</td>
<td>1</td>
</tr>
</tbody>
</table>
Adding longer circuits makes the clusters more distinguishable.

Longer GST circuits amplify noise, making the clusters more distinguishable.

We can use this structure to do **classification**!

(An independent PCA was done for each L.)
Classification is possible because the data sets cluster based on noise type and strength!

**Project** feature vectors based on **PCA**

**Label** feature vectors based on **noise**

**Train a soft-margin, linear support vector machine (SVM)**
Classification is possible because the data sets cluster based on noise type and strength!

Project feature vectors based on **PCA**

Label feature vectors based on **noise**

Train a soft-margin, linear support vector machine (**SVM**)

96% accuracy??

Cross-validation required!
Under cross-validation, the SVM has reasonably low inaccuracy.

SVM is fairly accurate - largest inaccuracy ~2%

20-fold shuffle-split cross-validation (25% withheld for testing)
The accuracy of the SVM is affected by the number of components and maximum sequence length.

<table>
<thead>
<tr>
<th>Noise Type</th>
<th>Number of Feature Vectors</th>
<th>Number of Noise Strengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coherent Error</td>
<td>450</td>
<td>9</td>
</tr>
<tr>
<td>Depolarization</td>
<td>450</td>
<td>9</td>
</tr>
<tr>
<td>No Noise</td>
<td>50</td>
<td>1</td>
</tr>
</tbody>
</table>

20-fold shuffle-split cross-validation scheme used, with 25% of the data withheld for testing on each split. A “one-versus-one” multi-class classification scheme was used.
Can a classifier learn the difference between arbitrary \textit{stochastic} and arbitrary \textit{coherent} noise?

\begin{align*}
\dot{\rho} &= -i[H_0, \rho] \\
&\quad - i[e, \rho] \\
\mathcal{E} &= V \circ G_0 \\
VV^T &= I
\end{align*}
Classification in a 2-dimensional subspace is harder, due to structure of PCA-projected feature vectors.

“Radio dish” type structure

Linear classifier infeasible with only 2 PCA components
Preliminary results indicate a linear, soft-margin SVM can classify these two noise types in higher dimensions.

For each $L$:
- 10 values of noise strength in $[10^{-4}, 10^{-1}]$
- 260 random instances

20-fold shuffle-split cross-validation scheme used, with 25% of the data withheld for testing on each split. A “one-verus-one” multi-class classification scheme was used.

Gap goes away if noise $\leq 10^{-2}$ removed from data
Specific machine learning tools can analyze GST circuits and learn about noise.
Specific machine learning tools can analyze GST circuits and learn about noise.
There are lots of problems at the intersection of device characterization and machine learning!
Thank you!

@Travis_Sch