Quantum error correction for the toric code using deep reinforcement learning

Mats Granath
Department of Physics
University of Gothenburg

Machine Learning for Quantum Technology
Max Planck Institute for the Science of Light
May 9th, 2019

Mattias Eliasson, David Fitzek, MG, in progress
Bottom line

We do the “simplest” error correction problem for a topological code
• Periodic boundary conditions
• No measurement noise/perfect syndrome
• only bit flip noise (initially, can also do depolarizing)

Still challenging for reinforcement learning: deep Q-networks needed
  Allows for easy benchmark
Fault-tolerant quantum computation by anyons

A. Yu. Kitaev*
L. D. Landau Institute for Theoretical Physics, 117940, Kusygin St. 2, Russia
Received 20 May 2002

The toric code

Topological quantum memory

Eric Dennis
Princeton University, Princeton, New Jersey 08544
Alexei Kitaev, Andrew Landahl, and John Preskill
Institute for Quantum Information, California Institute of Technology, Pasadena, California 91125

$H = -\sum_\alpha \hat{P}_\alpha - \sum_\nu \hat{V}_\nu$

$\hat{P}_\alpha = \prod_{i \in \alpha} \sigma_i^z$

$\hat{V}_\nu = \prod_{i \in \nu} \sigma_i^x$

Plaquette and Vertex stabilizers (parity checks)

$2d^2$ physical qubits, $2d^2-2$ independent stabilizers

Ground state

consider:

act with vertex op:

act with two vertex op:

GS is symmetric superposition of all trivial loops:

$|\text{plaq. op. ground state}\rangle$

still a plaquette ground state

still a plaquette ground state

$|\text{GS}_0\rangle = \sum_{i \text{all trivial loops}} \text{loop}_i |\uparrow\uparrow\uparrow \cdots\rangle$

highly entangled
Ground state degeneracy

Non-trivial loops (encircling torus) $X_1, X_2$ are not products of vertex operators.

Four ground states/The logical qubit

$$\{|GS_0\rangle, X_1|GS_0\rangle, X_2|GS_0\rangle, X_2X_1|GS_0\rangle\}$$

Distinguished by $\pm 1$ eigenvalues of $Z_1$ and $Z_2$.

Corresponding to $2(d^2-1)$ independent stabilizers on $2d^2$ physical qubits.

Topologically protected qubit

Non-trivial loops=Logical bit-flip operators

Requires at least $d$ physical bit-flip errors

**code distance** $d$
Error correction consider bit-flip errors

The syndrome (defects/bad plaquettes), is quantum non-demolition measurement

Ex.

two neighbouring bit flip errors, two defects

proper error correction trivial loop

failed error correction non-trivial loop

Standard algorithm to suggest error correcting strings:
Minimum Weight Perfect Matching (MWPM)/Blossom

J. Edmunds, 1965

Find shortest total correction string. (Which is the most likely)
Error models

Depolarizing

- (1-p) no error
- p/3  X
- p/3  Y=XZ
- p/3  Z

Uncorrelated

- (1-p)² no error  
  - p(1-p)  X  
  - p²  Y=XZ  
  - p(1-p)  Z  

Bit- and phase-flip errors (i.e. plaquette and vertex errors). are independent. Corrected separately.

MWPM is (near) optimal
Minimum Weight Perfect Matching Low-p fail rate for bit-flip errors

For $p \to 0$ we only need to consider error chains with minimal number of errors that can give failed error correction.

Consider $d=5$:

- Two errors is always corrected successfully.
- Three errors in a row always gives failed error correction.
- Three errors not in a row always gives successful correction.

MWPM asymptotic (lowest order in $p$) fail rate is:

$$p_L = 2d \left( \frac{d}{\lceil d/2 \rceil} \right) p^{d/2}$$
Deep reinforcement learning/Deep Q-learning

Human-level control through deep reinforcement learning

Volodymyr Mnih\textsuperscript{1}, Koray Kavukcuoglu\textsuperscript{1}, David Silver\textsuperscript{1}, Andrei A. Rusu\textsuperscript{1}, Joel Veness\textsuperscript{1}, Marc G. Bellemare\textsuperscript{2}, Alex Graves\textsuperscript{1}, Martin Riedmiller\textsuperscript{3}, Andreas K. Fidjeland\textsuperscript{4}, Georg Ostrovski\textsuperscript{1}, Stig Petersen\textsuperscript{1}, Charles Beattie\textsuperscript{1}, Amie Hatik\textsuperscript{1}, Ioannis Antonoglou\textsuperscript{1}, Helen King\textsuperscript{1}, Dhruvan Kamra\textsuperscript{2}, Daan Wierstra\textsuperscript{1}, Shane Legg\textsuperscript{1} & Demis Hassabis\textsuperscript{1}

Mastering the game of Go without human knowledge

David Silver\textsuperscript{1*}, Julian Schrittwieser\textsuperscript{1*}, Karen Simonyan\textsuperscript{3*}, Ioannis Antonoglou\textsuperscript{1}, Aja Huang\textsuperscript{1}, Arthur Guez\textsuperscript{1}, Thomas Hubert\textsuperscript{1}, Lucas Baker\textsuperscript{1}, Matthew Lai\textsuperscript{1}, Adrian Bolton\textsuperscript{1}, Yutian Chen\textsuperscript{1}, Timothy Lillicrap\textsuperscript{1}, Fan Hui\textsuperscript{1}, Laurent Sifre\textsuperscript{1}, George van den Driessche\textsuperscript{1}, Thore Graepel\textsuperscript{1} & Demis Hassabis\textsuperscript{1}

AlphaStar 2019
Q-learning

- Agent in an environment described by a state $s$.
- Agent takes actions $a$ to move between states, $s \rightarrow s'$.
- Reward (positive or negative) $r$ is given depending on state/action.
- Agent learns policy, $\pi(s,a)$, to navigate environment for optimal accumulated reward (return) by exploring.

Q-function (action-value fcn) $Q(s,a)$ quantifies expected return from taking action $a$ in state $s$ and subsequently following the optimal policy.

$$Q(s, a) = r + \gamma \max_{a'} Q(s', a')$$

$\gamma < 1$ is discounting factor, better to get reward now than later

Explore to get reward and learn $Q =>$ optimal policy

Difficult if big world with many states and actions

Use Artificial Neural Network to represent Q-function

Deep Q-learning
Q-learning for the toric code

**state** is a syndrome
**action** is a bitflip=cardinal move of defect
**reward**, $r=-1$ per move (i.e. we aim to learn MWPM)

State space is very big
number of ways of placing $N_S$ defects on $d^2$ sites:

$$\binom{d^2}{N_S} \approx \binom{49}{20} \sim 10^{13}$$

for $d=7$ and $p=10\%$

Use deep Q-learning
Efficient implementation of Q-network

Use translational and rotational symmetry to center each defect.

Syndrome

Observation
Perspective 1 Perspective 2 Perspective 3 Perspective 4

Convolutional NN

Mats Granath, MLQT, Erlangen 2019
Deep Q-network

Network gives Q-values for the 4 movements of the central defect. Crucial simplification, fixed number (4) actions, and doesn’t have to learn about boundaries.

Table 2: Network architecture \(d=7\).

<table>
<thead>
<tr>
<th>#</th>
<th>Type</th>
<th>Size</th>
<th># parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Input</td>
<td>7x7</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Conv.</td>
<td>512 filters; 3x3 size; 2-2 stride</td>
<td>5 120</td>
</tr>
<tr>
<td>2</td>
<td>FC</td>
<td>256 neurons</td>
<td>1 179 904</td>
</tr>
<tr>
<td>3</td>
<td>FC</td>
<td>128 neurons</td>
<td>32 896</td>
</tr>
<tr>
<td>4</td>
<td>FC</td>
<td>64 neurons</td>
<td>8 256</td>
</tr>
<tr>
<td>5</td>
<td>FC</td>
<td>32 neurons</td>
<td>2 080</td>
</tr>
<tr>
<td>6</td>
<td>FC (out)</td>
<td>4 neurons</td>
<td>132</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 228 388</td>
</tr>
</tbody>
</table>

Experience replay is crucial for training

Significant reduction in number of parameters. Size of state space for \(d=7\), and \(N_s=20\) defects (10% error)

\[
\left( \frac{d^2}{N_s} \right) \approx \left( \frac{49}{20} \right) \approx 10^{13}
\]
Results. Converged Q-network.

Examples:

Large arrow=Large Q-value for that action

Large arrow=Large Q-value for that action

\[ R = -1 - \gamma - \gamma^2 - \gamma^3 = -3.62 \]

\[ \gamma = 0.95 \]

(semi-) quantitatively correct Q-values
Philip Andreasson, Joel Johansson, Simon Liljestrand, Mats Granath, arXiv:1811.12338

Consider first the case of code distance $d$.

The reinforcement learning agent makes use of a deep neural network decoder.

A Small error rate $p_L$ fits asymptotic form for small $p$:

$$p_L = 2d \left( \frac{d}{[d/2]} \right) p^{[d/2]}$$
Depolarizing noise, work in progress

Example syndrome

MWPM

Reinforcement trained solver
reward=annihilation of complete syndrome + small intermediate reward

The agent can use Y to take advantage of correlations between bit-flip and phase-flip errors

Mats Granath, MLQT, Erlangen 2019
Preliminary performance of RL solver for depolarizing noise

Outperforms MWPM

Asymptotic behaviour for d=5 and d=7
<table>
<thead>
<tr>
<th>Layer (type)</th>
<th>Output Shape</th>
<th>Param #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d-1</td>
<td>[-1, 128, 5, 5]</td>
<td>2,432</td>
</tr>
<tr>
<td>Conv2d-2</td>
<td>[-1, 128, 5, 5]</td>
<td>147,584</td>
</tr>
<tr>
<td>Conv2d-3</td>
<td>[-1, 120, 5, 5]</td>
<td>138,360</td>
</tr>
<tr>
<td>Conv2d-4</td>
<td>[-1, 111, 5, 5]</td>
<td>119,991</td>
</tr>
<tr>
<td>Conv2d-5</td>
<td>[-1, 104, 5, 5]</td>
<td>104,000</td>
</tr>
<tr>
<td>Conv2d-6</td>
<td>[-1, 103, 5, 5]</td>
<td>96,511</td>
</tr>
<tr>
<td>Conv2d-7</td>
<td>[-1, 90, 5, 5]</td>
<td>83,520</td>
</tr>
<tr>
<td>Conv2d-8</td>
<td>[-1, 80, 5, 5]</td>
<td>64,880</td>
</tr>
<tr>
<td>Conv2d-9</td>
<td>[-1, 73, 5, 5]</td>
<td>52,633</td>
</tr>
<tr>
<td>Conv2d-10</td>
<td>[-1, 71, 5, 5]</td>
<td>46,718</td>
</tr>
<tr>
<td>Conv2d-11</td>
<td>[-1, 64, 3, 3]</td>
<td>40,960</td>
</tr>
<tr>
<td>Linear-12</td>
<td>[-1, 3]</td>
<td>1,731</td>
</tr>
</tbody>
</table>

Total params: 899,320

**distance 7 code**

<table>
<thead>
<tr>
<th>Layer (type)</th>
<th>Output Shape</th>
<th>Param #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d-1</td>
<td>[-1, 200, 7, 7]</td>
<td>3,800</td>
</tr>
<tr>
<td>Conv2d-2</td>
<td>[-1, 190, 7, 7]</td>
<td>342,190</td>
</tr>
<tr>
<td>Conv2d-3</td>
<td>[-1, 189, 7, 7]</td>
<td>323,379</td>
</tr>
<tr>
<td>Conv2d-4</td>
<td>[-1, 160, 7, 7]</td>
<td>272,320</td>
</tr>
<tr>
<td>Conv2d-5</td>
<td>[-1, 150, 7, 7]</td>
<td>216,150</td>
</tr>
<tr>
<td>Conv2d-6</td>
<td>[-1, 132, 7, 7]</td>
<td>178,332</td>
</tr>
<tr>
<td>Conv2d-7</td>
<td>[-1, 128, 7, 7]</td>
<td>152,192</td>
</tr>
<tr>
<td>Conv2d-8</td>
<td>[-1, 120, 7, 7]</td>
<td>138,360</td>
</tr>
<tr>
<td>Conv2d-9</td>
<td>[-1, 111, 7, 7]</td>
<td>119,991</td>
</tr>
<tr>
<td>Conv2d-10</td>
<td>[-1, 104, 7, 7]</td>
<td>104,000</td>
</tr>
<tr>
<td>Conv2d-11</td>
<td>[-1, 103, 7, 7]</td>
<td>96,511</td>
</tr>
<tr>
<td>Conv2d-12</td>
<td>[-1, 90, 7, 7]</td>
<td>83,520</td>
</tr>
<tr>
<td>Conv2d-13</td>
<td>[-1, 80, 7, 7]</td>
<td>64,880</td>
</tr>
<tr>
<td>Conv2d-14</td>
<td>[-1, 73, 7, 7]</td>
<td>52,633</td>
</tr>
<tr>
<td>Conv2d-15</td>
<td>[-1, 71, 7, 7]</td>
<td>46,718</td>
</tr>
<tr>
<td>Conv2d-16</td>
<td>[-1, 64, 5, 5]</td>
<td>40,960</td>
</tr>
<tr>
<td>Linear-17</td>
<td>[-1, 3]</td>
<td>4,803</td>
</tr>
</tbody>
</table>

Total params: 2,240,739

**Unnecessarily deep?**

trained on desktop GPU for 5 hours (using PyTorch)

trained on desktop GPU for 12 hours

Mats Granath, MLQT, Erlangen 2019
Conclusions

Deep Q-learning works well for error correction on toric code. Can match or even outperform MWPM (for moderate code distance)

But, does require quite deep Q-networks

Periodic boundaries important for our approach.

Future challenges:

- Larger code distances
- Improve reward scheme, use actual success or failure of error correction
- Include syndrome measurement error. (R. Sweke et al, arXiv:1810.07207)
- Surface code with boundaries. (Tougher due to lack of translational invariance)

Philip Andreasson, Joel Johansson, Simon Liljestrand, Mats Granath, arXiv:1811.12338
Mattias Eliasson, David Fitzek, MG, in progress