Quantum generalisations of feedforward neural nets
Oscar Dahlsten  |  dahlsten@sustech.edu.cn  |  Fort Canning, Singapore, Jan 2020

Joint work with: Kwok-Ho Wan, Hler Kristjansson, Robert Gardner, Myungshik Kim, Feiyang Liu, Yidong Liao and Daniel Ebler (chronologically ordered)
SUSTech aim: leading international research uni ‘Shenzhen’s (China’s Silicon Valley) Stanford’.

One of world’s largest quantum efforts:
Shenzhen Institute for Quantum Science and Engineering (SIQSE)

Peng Cheng (Eagle City) lab
Information science research institute, to be university sized.
‘Quantum generalisations of feedforward neural nets’

• *Neural nets* take inputs (e.g. bit strings) and process them, giving some output (e.g. another bit string).

• Their elementary components are called (artificial) *neurons*; these are connected, forming a *network* (net).

• *Feedforward* neural nets pass the data in one direction only during processing, e.g. left to right, as opposed to back and forth.

• By *quantum generalization* we mean neural nets which can process superpositions of bit strings without collapsing the superposition.
Motivation and aim

• Neural nets have proven strikingly versatile in the classical case, especially those of many layers (deep networks).
• Will quantum neural nets similarly benefit quantum information processing?
• Other underlying motivation: how to model quantum Maxwell’s demons.

• **Aim:** *design quantum neural nets* and *investigate their usefulness*, both for classical and quantum tasks.

### Aim to present key ideas of 3 papers

<table>
<thead>
<tr>
<th>Paper title</th>
<th>Authors</th>
<th>Reference</th>
<th>Network type</th>
<th>Training</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning Simon’s Algorithm</td>
<td>Wan, Dahlsten*, Liu, Kim</td>
<td>arXiv: 1806.10448</td>
<td>Same as above</td>
<td>Same as above</td>
</tr>
<tr>
<td>Quantum advantage for training binary neural nets</td>
<td>Liao, Ebler, Liu, Dahlsten*</td>
<td>arXiv: 1810.12948 (v2 Dec 2019)</td>
<td>Quantum unitaries controlled by quantum weights</td>
<td>Quantum search over weights</td>
</tr>
</tbody>
</table>
I: Quantum generalisations of feedforward neural nets
Overview

• Generalising neuron to quantum.

• Generalising network to quantum.

• Generalising training to quantum.

• Evaluating usefulness via examples:
  1. Rediscovering quantum teleportation
  2. Quantum autoencoder

• Physical implementation.

• Summary and outlook
Classical Neuron

\[ \text{out} = \begin{cases} 1 & \text{if } w_1 \text{in}_1 + w_2 \text{in}_2 > 0.5 \\ 0 & \text{else} \end{cases} \]

\( \text{in}_1, \text{in}_2 \) and \( \text{out} \) are bits here, \( w_1 \) and \( w_2 \) are real numbers.
Irreversible $\rightarrow$ reversible: For an $n$-input classical neuron taking 

$$(in_1, in_2) \rightarrow out,$$

create a classical logically reversible gate 

$$(in_1, in_2, 0) \rightarrow (in_1, in_2, out).$$

This can be represented by a permutation matrix $P$. 
**Generalising classical neuron, step 2**

**Reversible → unitary**

- Matrix $P$ is generalised to a unitary $U$ with complex entries.
- $U$ acts on quantum state, a vector with complex entries, associated with qubits $in1$, $in2$ and dummy input in state $|0\rangle$. 

![Diagram of quantum gates](image_url)
Generalising network: copying output → 2 qubit gate

Reversible version of classical copying is CNOT: 00 → 00, 10 → 11

We generalise CNOT to general ‘fan-out’ 2 qubit unitary $U_f$ (with one dummy input $|0\rangle$)
Generalising network, example

Quantum network contains classical as a special case

Experiments can use fewer qubits, (then not strict generalisation of classical)
Generalising the training: how does classical training work

- Basic idea of training: tune the network’s free parameters until it does what you want.

- You feed network inputs, check how often output is the desired output, quantifying performance with cost function $C$. Often

\[
C = \sum_{\text{training inputs } i} (\text{out}_i - \text{out}_{\text{desired}_i})^2,
\]

- To tune free parameter $\theta$, numerically evaluate $C(\theta)$, $C(\theta + \delta)$ and update as

\[
\theta \rightarrow \theta - \eta \frac{C(\theta + \delta) - C(\theta)}{\delta},
\]

where $\eta$ is a jump size parameter. Repeat until $C$ appears minimised. (This is called gradient descent)
Quantum case: Training via cost function and gradient descent

- Neuron Unitaries free parameters are tuned.
- Task represented by cost function.
- Classical output case: compare statistics in one measurement basis with desired statistics.
- Quantum output case: compare statistics on several measurements (pending on task), or do so-called SWAP test with desired output state (*Ackn. Aephraim Steinberg*).
- Gradient descent used to minimise cost.

For swap test approach see arxiv:1902.10445
*Efficient learning for deep quantum neural nets*, Beer et al
Evaluation of usefulness: can rediscover teleportation

Network successfully trained to find protocol to transfer state from Alice to Bob, using decohered 2 bit communication and one entangled state.

CAN YOU THINK OF NEW TASKS YOU WANT TO FIND PROTOCOLS FOR?
Autoencoders find encodings of data, from a given source, onto fewer bits. Widely used in classical machine learning.

To succeed, network must compress data through bottleneck.
Evaluation of usefulness: can realise quantum autoencoder


Experiment: A quantum autoencoder: the compression of qutrits via machine learning
Alex Pepper, Nora Tischler, Geoff J. Pryde
arxiv:1810.01637
Physical implementation

Can tune phase shifters in photonic integrated circuit using heaters.

Classical computer uses measurement data from photodiodes to tune the phases.

This type of design only works for small nets, need e.g. non-linear optics otherwise.
II: Learning Simon’s Algorithm
-can we discover quantum speed-ups through gradient descent training of unitaries?

classical data $\rightarrow$ $\psi$ $\rightarrow$ classical data
Simon’s algorithm to learn about black box $f(x)$

Simon's algorithm uses quantum interference to learn a property `s` of a black box function $f(x)$. Uses exponentially (in nr of input bits) fewer calls to $U_f$ than classical method to learn $s$.

![Diagram of quantum circuit for Simon's algorithm]

Create superposition of inputs $x$

We want to know which $U_f$ it is

Further input branching enables interference between terms which had different $x$.

$$U_f \sum_x |x, 0\rangle = \sum_x |x, f(x)\rangle$$
Learning Simon’s algorithm

- Given different $U_f$’s, the task is to output a guess for $s$
- Perform gradient descent to train the unitaries.
- **Conclusion**: Training converges to Simon’s circuit on previous slide, as hoped.
II: Quantum training of binary neural nets
Ackn. Prof Yu Hao, SUSTech EEE, for telling us binary neurons important

\[ a_1, a_2, w_2, w_3 \text{ and } \text{out} \text{ are all binary here, leading to simple gate implementation in hardware} \]

\[ \begin{array}{c}
  a_1 \\
  w_1 \\
  a_2 \\
  w_2 \\
\end{array} \quad \begin{array}{c}
  XNOR \\
  S_1 \\
  \text{Bit counter} \\
  S \\
\end{array} \quad \rightarrow \text{Sign}(S) = a' \]
Quantum generalisation of binary neural net

- Again take the route of making neurons reversible (R), then quantum (Q).
- This gives a quantum binary neuron, with qubits representing the weights.
- Experimentally, tuning weights now means tuning input states (not unitary gates)
Quantum method for training binary neural net

\[ \sum_i |w_i\rangle \quad U_{\text{neuron}} \quad U_{\text{neuron}}^{-1} \quad \sum_i e^{i\Delta \theta \Lambda(a_i, a^*)} |w_i\rangle \]

- Put the weights in a superposition rather than feed them one at a time.
- ‘Oracle’ unitary adds a phase \( \exp(i\Delta \theta) \) if output matches desired output.
- The weights are decoupled by uncomputing unitary, phases converted to \( \pm 1 \) by use of phase estimation and then a Grover search can be performed, singling out best weight state(s).
- Provably finds best weight string faster than classical brute force:
  \( \text{no}(\text{weight states}) \text{no}(\text{input states}) \rightarrow \sqrt{\text{no}(\text{weight states})\text{no}(\text{input states})} \)
Quantum method for training binary neural net: Huawei HiQ

- We implement training on Huawei HiQ.
- Amplitude of good weights get amplified by quantum search.
- Example on right for 5 neuron network (19 qubits).
- Also did 6 neurons (23 qubits) on HiQ.
- Scalable code made and will be made available on HiQ.

```
input_1 --> input_2
```

```
output
```

```
```
### Summary and outlook

<table>
<thead>
<tr>
<th>Paper title</th>
<th>Authors</th>
<th>Reference</th>
<th>Main message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quantum generalization of feedforward neural nets</td>
<td>Wan, Dahlsten*, Kristjánsson, Gardner, Kim</td>
<td>npj Quantum Information 3, 36 (2017).</td>
<td>Classical neural nets are tuneable logical gates, quantum ones can be defined as generalisation: tuneable unitaries. Two uses (i) quantum tasks (e.g. discovering QI protocols, compressing superpositions ), (ii) classical tasks-see below.</td>
</tr>
<tr>
<td>Learning Simon’s Algorithm</td>
<td>Wan, Dahlsten*, Liu, Kim</td>
<td>arXiv: 1806.10448</td>
<td>Tuning unitaries around a black box function, network found an exponential quantum speed-up for classical task of probing black box, using superposition of inputs.</td>
</tr>
<tr>
<td>Quantum advantage for training binary neural nets</td>
<td>Liao, Ebler, Liu, Dahlsten*</td>
<td>arXiv: 1810.12948 (v2 to appear next week)</td>
<td>Can use quantum search to find best weight string, leading to quadratic speed-up in training relative to brute force.</td>
</tr>
</tbody>
</table>

We are hiring at all levels

THANK YOU