An Introduction to Deep Learning and its Future Development

19th - 20th September 2018 | AlgoUK - Liverpool
Dr Thomas Gorry
About Me

Born in Liverpool

PhD in Computer Science from The University of Liverpool, 2015

Founded Quanovo in 2015

MD and Chief Machine Learning Scientist

Twitter: @TGorry
E-Mail: t.gorry@quanovo.com
What does Quanovo do?

Data Management and Storage
- Relational Databases
- Non-Relational Databases
- Data Lakes

Data Sources
- Apps
- Sensors and devices

Data

Machine Learning and Analytics
- Descriptive Analytics
- Predictive Analytics
- Prescriptive Analytics

Intelligence
- Cognitive Service APIs
- Data Storage
- Power BI
- Custom

Intelligence

Dashboards & Visualizations
- Apps
- Mobile
- Bots

Action

People

Automated Systems

Web
<table>
<thead>
<tr>
<th>Traditional algorithms</th>
<th>regression analysis, markov processes, statistical inference, linear classifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specialised algorithms</td>
<td>page rank, collaborative filtering, dimensionality reduction, Bayesian models, …</td>
</tr>
<tr>
<td>ML and DL algorithms</td>
<td>ANN: feed-forward DNN, RNN, LSTM, convolutional, capsule, …</td>
</tr>
</tbody>
</table>
Our Partners
Deep Learning
Classification
Classification

[Diagram showing a classification process]
Classification

- Solid
- Vertical
- Diagonal
- Horizontal
Classification

1. Detect Inputs
2. Collect Inputs
3. Produce Output
4. Map Output to Target (nerve, muscle, gland)

Information Flow

Solid
Vertical
Diagonal
Horizontal
Input Neurons
Pixel Brightness

-1.0  -0.75  -0.50  -0.25  0.0  +0.25  +0.50  +0.75  +1.0
Receptive Fields

-0.75
-0.50
0.0
0.75
Adding a neuron

[Diagram showing a neuron with connections and weights: 0.50, 0.0, -0.75, -0.75, 0.75]
Summing up inputs

.50
0.0
-.75
.75

.50

+

Input nodes connected to a sum node with values .50, 0.0, -.75, and .75.
Weights

Diagram showing weights connected to nodes with values 0.0, 0.75, and 0.50.
Weights

0.50
0.0
-0.75
0.8
-1.07
5

0.0
-0.2
-0.5

0.75
Weights
Squashing
Sigmoid Squashing Function
Collapsed Neuron

Diagram showing a neuron with inputs and weights: 0.50, 0.0, 0.0, -0.75, 0.8, -0.5. The output of the neuron is calculated as 5(-1.07) + 0.746.
Collapsed Neuron

![Diagram of a collapsed neuron with values: -0.75, 0.0, 0.50, 0.75, 0.746.](image)
Many Neurons
Many Neurons

Weights

- 1.0
- -1.0
- 0
Receptive Fields

Weights

- 1.0
- -1.0
- 0
Additional Layers

Weights
- 1.0
- -1.0
- 0
Receptive Fields

Weights

- 1.0
- -1.0
- 0
New Neuron Type

Weights
- 1.0
- -1.0
- 0
Rectified Linear Unit (ReLu)
New Neuron Type

Weights
- 1.0
- -1.0
- 0
Output Layer

Solid
Vertical
Diagonal
Horizontal
Run-through

Solid
Vertical
Diagonal
Horizontal
Run-through
Run-through
Error

<table>
<thead>
<tr>
<th>Truth</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Solid</td>
</tr>
<tr>
<td>0</td>
<td>Vertical</td>
</tr>
<tr>
<td>0</td>
<td>Diagonal</td>
</tr>
<tr>
<td>1</td>
<td>Horizontal</td>
</tr>
<tr>
<td>Truth</td>
<td>Answer</td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
</tr>
<tr>
<td>0</td>
<td>.5</td>
</tr>
<tr>
<td>0</td>
<td>.75</td>
</tr>
<tr>
<td>0</td>
<td>-.25</td>
</tr>
<tr>
<td>1</td>
<td>-.75</td>
</tr>
<tr>
<td>Error</td>
<td>Truth</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>.5</td>
<td>0</td>
</tr>
<tr>
<td>.75</td>
<td>0</td>
</tr>
<tr>
<td>.25</td>
<td>0</td>
</tr>
<tr>
<td>1.75</td>
<td>1</td>
</tr>
<tr>
<td>Error</td>
<td>Truth</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>.5</td>
<td>0</td>
</tr>
<tr>
<td>.75</td>
<td>0</td>
</tr>
<tr>
<td>.25</td>
<td>0</td>
</tr>
<tr>
<td>1.75</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>3.25</td>
</tr>
</tbody>
</table>
Learning Weights
Computationally Expensive
Calculating the Slope

Slope = $\frac{\delta e}{\delta w}$

Error At

Original Weight

Weight
Calculating the Slope

Slope = $\frac{\delta e}{\delta w}$

$-2/+1 = -2$
Calculating the Slope

Slope = \( \frac{\delta e}{\delta w} \)

Error Function = \( w^{12} \)

\[
\frac{\delta e}{\delta w} \times w = 2 \times -1 = -2
\]
Chaining

\[ y = x \times w_1 \]
\[ \frac{\delta y}{\delta w_1} \]

\[ e = y \times w_2 \]
\[ \frac{\delta e}{\delta y} = 2 \]

\[ e = x \times w_1 \times w_2 \]
\[ \frac{\delta e}{\delta w_1} \times w_2 \]
\[ \frac{\delta e}{\delta w_1} \times w_1 \]
\[ \frac{\delta e}{\delta y} \]
Backpropagation

\[
\frac{\delta \text{error}}{\delta \text{weight}} \quad \frac{\delta a}{\delta \text{weight}} \times \frac{\delta b}{\delta a} \times \frac{\delta c}{\delta b} \times \frac{\delta d}{\delta c} \times \ldots \times \frac{\delta y}{\delta x} \times \frac{\delta z}{\delta y} \times \frac{\delta \text{error}}{\delta z}
\]
Backpropagation Elements - Weights

\[
\frac{\delta \text{error}}{\delta a} \quad \frac{\delta b}{\delta b} \\
\frac{\delta \text{error}}{\delta b}
\]

\[b = \text{weight} \times a\]

\[\frac{\delta b}{\delta a}\]
Backpropagation Elements - Sums

\[ \frac{\delta error}{\delta a} \quad \frac{\delta z}{\delta a} \]
\[ \frac{\delta error}{\delta z} \]

\[ z = a + b + c + d + \ldots \]

\[ \frac{\delta z}{\delta a} \]
Backpropagation Elements - Sigmoid

\[
\frac{\delta \text{error}}{\delta a} \quad \frac{\delta b}{\delta \hat{b}} \\
\frac{\delta \text{error}}{\delta \hat{b}}
\]

\[
b = \frac{1}{1 + e^{a}} \quad \sigma(a)
\]

\[
\frac{\delta b}{\delta a} \quad \sigma(\hat{b}) \ast (1 - \sigma(a))
\]
Backpropagation Elements - ReLu

\[
\frac{\delta \text{error}}{\delta a} \quad \frac{\delta b}{\delta a} \\
\frac{\delta \text{error}}{\delta b}
\]

\[
b \quad = \quad a > 0 \\
= \quad 0, \text{ otherwise}
\]

\[
\frac{\delta b}{\delta a} \quad , \quad a > 0 \\
= \quad 0, \text{ otherwise}
\]
Training

Solid
Vertical
Diagonal
Horizontal
## Hyper-Parameters

<table>
<thead>
<tr>
<th>Mini-Batch Gradient Descent</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Learning Rate</td>
<td>• Hidden Units</td>
</tr>
<tr>
<td>• Loss function</td>
<td>• Weight Decay</td>
</tr>
<tr>
<td>• Mini-Batch Size</td>
<td>• Activation Sparsity</td>
</tr>
<tr>
<td>• Training Iterations</td>
<td>• Non-Linearity</td>
</tr>
<tr>
<td>• Momentum</td>
<td>• Weight Initialization</td>
</tr>
</tbody>
</table>
Use Cases
Applications of AI

- churn prediction
- demand forecasting
- fraud detection
- financial forecasting
- predictive maintenance
- recommender systems
- image and video recognition
- chat bots
- identifying cross-selling opportunities
- sentiment analysis
- natural language processing

*Future AI demand trajectory, % change in AI spending over next 3 years*

1. Estimated average, weighted by company size; demand trajectory based on midpoint of range selected by survey respondent.
2. Adopting 1 or more AI technologies at scale or in business core; weighted by company size.

*Source: McKinsey Global Institute AI adoption and use survey; McKinsey Global Institute analysis*
Legal Profession

- Discovery assistance
- Judge analysis
- Opposition scrutiny
- Case strategy advice
Financial & Insurance

- Fraud detection
- Cross selling
- Trading analytics
- Customer churn
- Risk pricing
Government & Local Authorities

- Housing fraud detection
- Recidivism forecasting
- Optimising waste collection routes
- Automated parking ticket system
Manufacturing

• Predictive Maintenance
• Stock tracking
Object Detection
Object Detection
Challenges
Challenges

**TECHNICAL**
- labelling
- over fitting
- data cleaning
- feature engineering
- model bias
- false correlation protection

**BUSINESS**
- data acquisition
- understanding limitations
- data security and privacy
- workforce fears / limited trained workforce
- AI mistrust / lack of explainability
- responsibility
Open Problems

Hyper-parameter Optimisation

<table>
<thead>
<tr>
<th>Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Not doing HPO can have a significant impact on model performance.</td>
</tr>
<tr>
<td>• Hand tuning is inefficient.</td>
</tr>
<tr>
<td>• Existing methods limited by speed and scalability.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Bayesian Optimisation is often used.</td>
</tr>
<tr>
<td>• Blending strategies is a common approach to improvement.</td>
</tr>
<tr>
<td>• Latest attempt:</td>
</tr>
<tr>
<td>\textit{Practical Hyperparameter Optimization for Deep Learning}</td>
</tr>
<tr>
<td>Falkner et al., 2018</td>
</tr>
</tbody>
</table>
## Open Problems

### Peering inside the Black Box

<table>
<thead>
<tr>
<th>Problem</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lack of transparency regarding why an output was achieved.</td>
<td><strong>Attention Based LSTM for Aspect-level Sentiment Classification</strong> – Sines a spotlight on where the model is looking when it makes a particular decision</td>
</tr>
</tbody>
</table>

Huang et Al, 2016

The fajita we tried was **tasteless** and **burned** and the **mole sauce** was **way too sweet**.

They have one of the **fastest delivery times** in the **city**.
Open Problems

Peering inside the Black Box

Problem
- Lack of transparency regarding why an output was achieved.

Status
LIME (Local-Interpretable-model-agnostic explanations) – Sensitivity analysis revealing the parts of an input that matter most to the eventual output.

Guestrin et Al, 2016

Tree Frog 54%
Billiard Balls 7%
Balloon 5%
### Open Problems

**Peering inside the Black Box**

<table>
<thead>
<tr>
<th>Problem</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lack of transparency regarding why an output was achieved.</td>
<td>LIME (Local-Interpretable-model-agnostic explanations) – Sensitivity analysis revealing the parts of an input that matter most to the eventual output. Guestrin et Al, 2016</td>
</tr>
<tr>
<td></td>
<td>• Good at showing which pixels in the image are most important for classification.</td>
</tr>
<tr>
<td></td>
<td>• Does not analyse how the pixel features are propagated through the network to arrive at a prediction.</td>
</tr>
</tbody>
</table>
Open Problems

Peering inside the Black Box

Problem
- Lack of transparency regarding why an output was achieved.

Status
APPLE: Automatic Patch Pattern Labelling for Explanation
Konam et Al, 2018

<table>
<thead>
<tr>
<th>Layer</th>
<th>Neuron</th>
<th>Patches</th>
<th>Predictions: ear, eye, fur, nose, paw, none</th>
</tr>
</thead>
<tbody>
<tr>
<td>Layer : 7</td>
<td>Neuron : 2</td>
<td><img src="image1" alt="Patches Image" /></td>
<td>0.56 0.27 0.06 0.02 0.04 0.05</td>
</tr>
<tr>
<td>Layer : 7</td>
<td>Neuron : 3</td>
<td><img src="image2" alt="Patches Image" /></td>
<td>0.20 0.51 0.05 0.08 0.06 0.10</td>
</tr>
<tr>
<td>Layer : 7</td>
<td>Neuron : 4</td>
<td><img src="image3" alt="Patches Image" /></td>
<td>0.08 0.22 0.50 0.03 0.10 0.07</td>
</tr>
<tr>
<td>Layer : 6</td>
<td>Neuron : 4</td>
<td><img src="image4" alt="Patches Image" /></td>
<td>0.08 0.25 0.50 0.02 0.10 0.05</td>
</tr>
<tr>
<td>Layer : 6</td>
<td>Neuron : 1</td>
<td><img src="image5" alt="Patches Image" /></td>
<td>0.08 0.24 0.49 0.03 0.10 0.06</td>
</tr>
</tbody>
</table>
# Open Problems

## Peering inside the Black Box

<table>
<thead>
<tr>
<th>Problem</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lack of transparency regarding why an output was achieved.</td>
<td>APPLE: Automatic Patch Pattern Labelling for Explanation</td>
</tr>
</tbody>
</table>

Konam et al., 2018

- Only works with images
- Does not scale well with many classes
### Open Problems

**Data Volume Requirement**

<table>
<thead>
<tr>
<th>Problem</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>A very large amount of data is needed to achieve good accuracy.</td>
<td>The goal is “One Shot Learning”.</td>
</tr>
<tr>
<td>Can be costly to obtain and end up excluding smaller businesses from the benefits of AI.</td>
<td>Only one data point is needed for reasonable accuracy.</td>
</tr>
<tr>
<td></td>
<td>Cheaper access to the technology.</td>
</tr>
<tr>
<td></td>
<td>Likely faster to train.</td>
</tr>
</tbody>
</table>
## Open Problems

### Data Volume Requirement

<table>
<thead>
<tr>
<th>Problem</th>
<th>Status</th>
</tr>
</thead>
</table>
| A very large amount of data is needed to achieve good accuracy. | CLEAR: Cumulative LEARning for One-Shot One-Class Image Recognition
Kozerawski et al, 2018 |
| Can be costly to obtain and end up excluding smaller businesses from the benefits of AI. | • Outperformed previous approaches on five benchmark datasets.  
• Limited to only one-class. |
## Open Problems

### Data Volume Requirement

<table>
<thead>
<tr>
<th>Problem</th>
<th>Status</th>
</tr>
</thead>
</table>
| A very large amount of data is needed to achieve good accuracy. | **Low-Shot Learning with Imprinted Weights**  
Qi et Al, 2018  
- Instant good classification performance on novel categories.  
- Tested with 1, 2, 5, 10 and 20 examples of each class.  
- 100 way classification  
- With 20 examples accuracy was at ~70%  
- With 1 example accuracy was only ~26% |
| Can be costly to obtain and end up excluding smaller businesses from the benefits of AI. |
Thank You