Life Long Learning

Hung-yi Lee

你用同一個腦學習
機器學習的每一堂課

但是每一個作業你都
訓練不同的類神經網路

能不能每次作業都訓練同一個類神經網路呢？
Life Long Learning (LLL)

Continuous Learning, Never Ending Learning, Incremental Learning

I can solve task 1.
I can solve tasks 1&2.
I can solve tasks 1&2&3.

Learning Task 1
Learning Task 2
Learning Task 3

...
Life-long Learning

Knowledge Retention
• but NOT Intransigence

Knowledge Transfer

Model Expansion
• but Parameter Efficiency
Example – Image

Task 1

This is “0”.

Task 2

This is “0”.

= 3 layers, 50 neurons each

<table>
<thead>
<tr>
<th>Task 1</th>
<th>Task 2</th>
<th>Forget!!!</th>
</tr>
</thead>
<tbody>
<tr>
<td>90%</td>
<td>96%</td>
<td>97%</td>
</tr>
<tr>
<td>80%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
明明可以把 Task 1 和 2 都學好，為什麼會變成這樣子呢!??
Example – Question Answering

• Given a document, answer the question based on the document.

• There are 20 QA tasks in bAbi corpus.
• Train a QA model through the 20 tasks

(a) QRN unit  (b) 2-layer QRN  (c) Overview

Task 5: Three Argument Relations
Mary gave the cake to Fred.
Fred gave the cake to Bill.
Jeff was given the milk by Bill.
Who gave the cake to Fred? A: Mary
Who did Fred give the cake to? A: Bill

Task 10: Indefinite Knowledge
John is either in the classroom or the playground.
Sandra is in the garden.
Is John in the classroom? A: maybe
Is John in the office? A: no

Task 15: Basic Deduction
Sheep are afraid of wolves.
Cats are afraid of dogs.
Mice are afraid of cats.
Gertrude is a sheep.
What is Gertrude afraid of? A: wolves
Example – Question Answering

Sequently train the 20 tasks

Jointly training the 20 tasks

感謝何振豪同學提供實驗結果

是不為也

非不能也
Catastrophic Forgetting
Wait a minute ......

- Multi-task training can solve the problem!

Using all the data for training → Computation issue

Training Data for Task 1

Training Data for Task 999

Training Data for Task 1000

Always keep the data → Storage issue

- Multi-task training can be considered as the upper bound of LLL.
Elastic Weight Consolidation (EWC)

Basic Idea: Some parameters in the model are important to the previous tasks. Only change the unimportant parameters. \( \theta^b \) is the model learned from the previous tasks.

Each parameter \( \theta^b_i \) has a “guard” \( b_i \)

\[
L'(\theta) = L(\theta) + \lambda \sum_i b_i (\theta_i - \theta^b_i)^2
\]

- Loss for current task
- Parameters to be learning
- Parameters learned from previous task
- How important this parameter is
- Loss to be optimized
Elastic Weight Consolidation (EWC)

Basic Idea: Some parameters in the model are important to the previous tasks. Only change the unimportant parameters. 

$\theta^b$ is the model learned from the previous tasks.

Each parameter $\theta_i^b$ has a “guard” $b_i$

One kind of regularization. $\theta_i$ should be close to $\theta^b$ in certain directions.

\[
L' (\theta) = L (\theta) + \lambda \sum_i b_i (\theta_i - \theta_i^b)^2
\]

If $b_i = 0$, there is no constraint on $\theta_i$

If $b_i = \infty$, $\theta_i$ would always be equal to $\theta_i^b$
Elastic Weight Consolidation (EWC)

The error surfaces of tasks 1 & 2.
(darker = smaller loss)
Elastic Weight Consolidation (EWC)

Each parameter has a “guard” $b_i$

Small 2nd derivative
- $b_1$ is small 動到沒關係

Large 2nd derivative
- $b_2$ is large 動到會出事
Elastic Weight Consolidation (EWC)

Task 1

\[ b_1 \text{ is small, while } b_2 \text{ is large.} \]

(可以動 \( \theta_1 \)，但儘量不要動到 \( \theta_2 \))

Do not forget!
Elastic Weight Consolidation (EWC)

MNIST permutation, from the original EWC paper
Elastic Weight Consolidation (EWC)

- Elastic Weight Consolidation (EWC)

- Synaptic Intelligence (SI)

- Memory Aware Synapses (MAS)
  - Special part: Do not need labelled data
Generating Data

- Conducting multi-task learning by generating pseudo-data using generative model

[Diagram showing the process of generating data for tasks 1 and 2, and solving the tasks.]
Adding New Classes

- Learning without forgetting (LwF)

- iCaRL: Incremental Classifier and Representation Learning
Life-long Learning

Knowledge Retention
• but NOT Intransigence

Knowledge Transfer

Model Expansion
• but Parameter Efficiency
Wait a minute ......

• Train a model for each task

➢ Knowledge cannot transfer across different tasks
➢ Eventually we cannot store all the models ...
Life-Long v.s. Transfer

Transfer Learning:
I can do task 2 because I have learned task 1.

(We don’t care whether machine can still do task 1.)

Life-long Learning:
Even though I have learned task 2, I do not forget task 1.
### Evaluation

$R_{i,j}$: after training task $i$, performance on task $j$

If $i > j$,
After training task $i$, does task $j$ be forgot

If $i < j$,
Can we transfer the skill of task $i$ to task $j$

<table>
<thead>
<tr>
<th>Rand Init.</th>
<th>Task 1</th>
<th>Task 2</th>
<th>......</th>
<th>Task T</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{0,1}$</td>
<td>$R_{0,2}$</td>
<td>( R_{0,T} )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>After Training</th>
<th>Task 1</th>
<th>Task 2</th>
<th>......</th>
<th>Task T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>( R_{1,1} )</td>
<td>$R_{1,2}$</td>
<td>( R_{1,T} )</td>
<td></td>
</tr>
<tr>
<td>Task 2</td>
<td>$R_{2,1}$</td>
<td>( R_{2,2} )</td>
<td>( R_{2,T} )</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task T-1</td>
<td>$R_{T-1,1}$</td>
<td>$R_{T-1,2}$</td>
<td>( R_{T-1,T} )</td>
<td></td>
</tr>
<tr>
<td>Task T</td>
<td>( R_{T,1} )</td>
<td>( R_{T,2} )</td>
<td>( R_{T,T} )</td>
<td></td>
</tr>
</tbody>
</table>

Accuracy = $\frac{1}{T} \sum_{i=1}^{T} R_{T,i}$

Backward Transfer = $\frac{1}{T-1} \sum_{i=1}^{T-1} R_{T,i} - R_{i,i}$

(It is usually negative.)
## Evaluation

$R_{i,j}$: after training task $i$, performance on task $j$

**If $i > j$,**

After training task $i$, does task $j$ be forgot

**If $i < j$,**

Can we transfer the skill of task $i$ to task $j$

<table>
<thead>
<tr>
<th>Rand Init.</th>
<th>Test on</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>$R_{0,1}$</td>
</tr>
<tr>
<td>Task 2</td>
<td>$R_{1,1}$</td>
</tr>
<tr>
<td>Task $T-1$</td>
<td>$R_{T-1,1}$</td>
</tr>
<tr>
<td>Task $T$</td>
<td>$R_{T,1}$</td>
</tr>
</tbody>
</table>

Accuracy = $\frac{1}{T} \sum_{i=1}^{T} R_{T,i}$

Backward Transfer = $\frac{1}{T-1} \sum_{i=1}^{T-1} R_{T,i} - R_{i,i}$

Forward Transfer = $\frac{1}{T-1} \sum_{i=2}^{T} R_{i-1,i} - R_{0,i}$
Gradient Episodic Memory (GEM)

- Constraint the gradient to improve the previous tasks

\[ g' \cdot g^1 \geq 0 \]
\[ g' \cdot g^2 \geq 0 \]

\[ g^1 \text{ : negative gradient of current task} \]
\[ g^2 \text{ : negative gradient of previous task} \]
\[ g' \text{ : update direction} \]

Need the data from the previous tasks
Life-long Learning

Knowledge Retention
• but NOT Intransigence

Knowledge Transfer

Model Expansion
• but Parameter Efficiency
Progressive Neural Networks

https://arxiv.org/abs/1606.04671
Expert Gate
https://arxiv.org/abs/1611.06194
Net2Net

Expand the network only when the training accuracy of the current task is not good enough.

Add some small noises

https://arxiv.org/abs/1511.05641

https://arxiv.org/abs/1811.07017
Concluding Remarks

Knowledge Retention

• but NOT Intransigence

Knowledge Transfer

Model Expansion

• but Parameter Efficiency
Curriculum Learning: what is the proper learning order?

Task 1

90%

Task 2

96%

Forget!!!

Task 1

80%

Task 2

97%

Task 2

97%

Task 1

62%

Task 2

90%

Task 1

97%
taskonomy
= task + taxonomy
(分類學)

http://taskonomy.stanford.edu/#abstract