A Small Step to Remember: Study of Single Model VS Dynamic Model

Liguang Zhou

School of Science and Engineering,
The Chinese University of Hong Kong, Shenzhen
Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS)

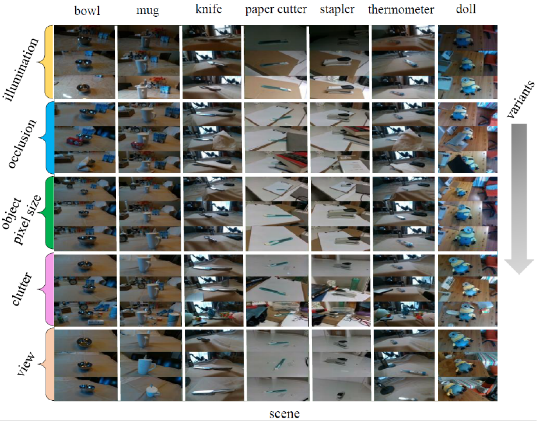
November 4, 2019

Overview

- Introduction
- Elastic Weights Consolidation (EWC) - Single Model
- Learning without Forgetting (LwF) - Dynamic Model
- Experiments
- Conclusion

Competition Details

Level	Illumination	Occlusion (percentage)	Object Pixel Size (pixels)	Clutter	Context	#Class	#Instance	#View
1	Strong	0%	$> 200 \times 200$	Simple	Home/office/mall	19	69	260
2	Normal	25%	$30 \times 30 - 200 \times 200$	Normal				
3	Weak	50%	$< 30 \times 30$	Complex				



Introduction

- In robotics area, the incremental learning of various objects is an essential problem for perception of robots.
- When there are many tasks to be trained in sequence, the DNNs will be suffering from catastrophic forgetting problem.
- One way to solve this catastrophic problem is called multi-task training, in which the various task will be trained concurrently in the training process. This solution can also be regarded as the upper bound of the Life Long Learning problem.
- However, in reality, if we need to train DNNs every time when new the task comes, it is low-efficiency and a lot of computing resources will be waste

Introduction

- In robotics area, the incremental learning of various objects is an essential problem for perception of robots.
- When there are many tasks to be trained in sequence, the DNNs will be suffering from catastrophic forgetting problem.
- One way to solve this catastrophic problem is called multi-task training, in which the various task will be trained concurrently in the training process. This solution can also be regarded as the upper bound of the Life Long Learning problem.
- However, in reality, if we need to train DNNs every time when new the task comes, it is low-efficiency and a lot of computing resources will be waste

Introduction

- Therefore, the alternative methods of solving this life long learning problem have been proposed, such as Elastic Weights Consolidation (EWC), Learning without Forgetting (LwF), generative methods and so on.
- EWC is a single model that utilize the Fisher Information Matrix, which is also related to the second derivative of the gradient, to preserve some important parameters of the previous tasks during the training.
- LwR is a dynamic model used for preserve the memory of the previous tasks by expend the network and introducing the knowledge distillation loss.

Elastic Weights Consolidation (EWC)

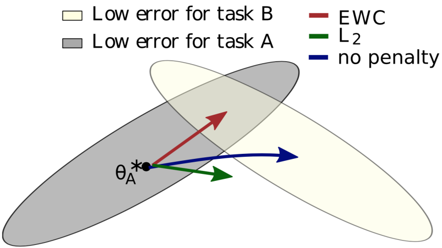


Figure 1: The learning sequence is from task A to task B

- We assume some parameters that are less useful and others are more valuable in DNNs. In the sequentially training, each parameter is treated equally. In EWC, we intend to utilize the diagonal components in Fisher Information Matrix to identify the importance of parameters to task A and apply the corresponding weights to them.

L2 Case

- To avoid forgetting the learned knowledge in task A, one simple trick is to minimize the distances between $\theta, \theta^*_{\mathcal{A}}$, which also can be regarded as L_2.

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \mathcal{L}_{\mathcal{B}}(\theta) + \frac{1}{2}\alpha \left(\theta - \theta^*_{\mathcal{A}}\right)^2 \tag{1}$$

- In L_2 case, each parameters is treated equally, which is not a wise solution because the sensitivity of each parameters varies a lot. The assumption is the importance of each parameters is different and varies a lot. Hence, the diagonal components in Fisher Information Matrix is used to measure the weights of importance of each parameter.

Close Look at EWC

- Baye's rule

$$\log p(\theta|\mathcal{D}) = \log p(\mathcal{D}|\theta) + \log p(\theta) - \log p(\mathcal{D}) \tag{2}$$

- Assume data is split into two parts, one defining task A (D_A) and the other defining task B (D_B), we obtain:

$$\log p(\theta|\mathcal{D}) = \log p\left(\left(\mathcal{D}_B|\theta\right) + \log p\left(\theta|\mathcal{D}_A\right) - \log p\left(\mathcal{D}_B\right) \tag{3}$$

- Fisher Information Matrix

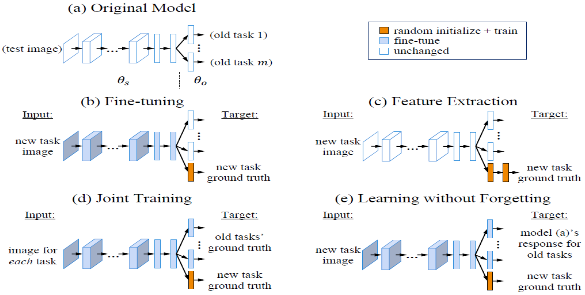
$$\theta^* = \underset{\theta}{\operatorname{argmin}} \mathcal{L}_B(\theta) + \frac{1}{2}\alpha F_{\theta_A^*,i}\left(\theta_i - \theta_{\mathcal{A},i}^*\right)^2$$

$$F_{\theta_A^*} = \frac{1}{N}\sum_{i=1}^{N} \nabla_\theta \log p\left(x_{\mathcal{A},i}|\theta_{\mathcal{A}}^*\right) \nabla_\theta \log p\left(x_{\mathcal{A},i}|\theta_{\mathcal{A}}^*\right)^T$$

- Loss function, L_B is the loss for task B only and λ indicates how important the old task is.

$$\mathcal{L}(\theta) = \mathcal{L}_B(\theta) + \sum_i \frac{\lambda}{2} F_i \left(\theta_i - \theta_{A,i}^*\right)^2 \tag{4}$$

Learning without Forgetting (LwF)



- θ_s: a set of shared parameters for CNNs (e.g., five convolutional layers and two fully connected layers for AlexNet [3] architecture)
- θ_0: task-specific parameters for previously learned tasks (e.g., the output layer for ImageNet [4] classification and corresponding weights)
- θ_n: randomly initialized task specific parameters for new tasks

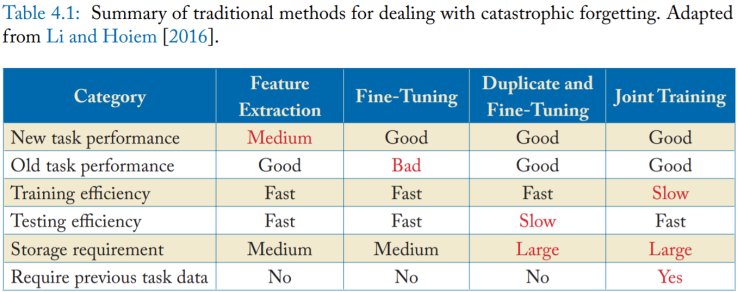
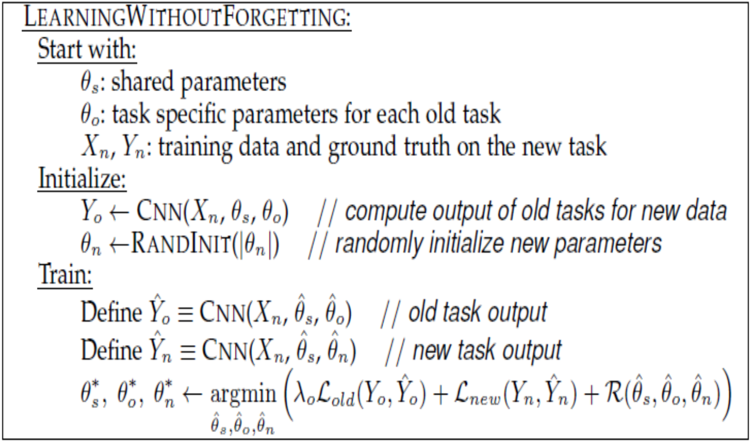


Table 4.1: Summary of traditional methods for dealing with catastrophic forgetting. Adapted from Li and Hoiem [2016].

Category	Feature Extraction	Fine-Tuning	Duplicate and Fine-Tuning	Joint Training
New task performance	Medium	Good	Good	Good
Old task performance	Good	Bad	Good	Good
Training efficiency	Fast	Fast	Fast	Slow
Testing efficiency	Fast	Fast	Slow	Fast
Storage requirement	Medium	Medium	Large	Large
Require previous task data	No	No	No	Yes

Close Look at LwR



```
LEARNINGWITHOUTFORGETTING:
Start with:
    $\theta_s$: shared parameters
    $\theta_o$: task specific parameters for each old task
    $X_n, Y_n$: training data and ground truth on the new task
Initialize:
    $Y_o \leftarrow \text{CNN}(X_n, \theta_s, \theta_o)$    // compute output of old tasks for new data
    $\theta_n \leftarrow \text{RANDINIT}(|\theta_n|)$    // randomly initialize new parameters
Train:
    Define $\hat{Y}_o \equiv \text{CNN}(X_n, \hat{\theta}_s, \hat{\theta}_o)$    // old task output
    Define $\hat{Y}_n \equiv \text{CNN}(X_n, \hat{\theta}_s, \hat{\theta}_n)$    // new task output
    $\theta_s^*, \theta_o^*, \theta_n^* \leftarrow \underset{\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n}{\operatorname{argmin}} \left( \lambda_o \mathcal{L}_{old}(Y_o, \hat{Y}_o) + \mathcal{L}_{new}(Y_n, \hat{Y}_n) + \mathcal{R}(\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n) \right)$
```

Figure 2: The details of algorithms

R: regularization term to avoid overfitting

- Loss function

$$\mathcal{L}_{new}\left(\mathbf{y}_n, \hat{\mathbf{y}}_n\right) = -\mathbf{y}_n \cdot \log \hat{\mathbf{y}}_n \tag{5}$$

- y_n is the one-hot ground truth label vector
- $\hat{y}_n$ is the softmax output of the network
- Knowledge Distillation loss

$$\begin{aligned}\mathcal{L}_{old}\left(\mathbf{y}_o, \hat{\mathbf{y}}_o\right) &= -H\left(\mathbf{y}'_o, \hat{\mathbf{y}}'_o\right) \\ &= -\sum_{i=1}^{l} y_o'^{(i)} \log \hat{y}_o'^{(i)}\end{aligned}$$

- l: number of labels
- $y_o^{(i)}$: ground truth/recorded probability
- $\hat{y}_o^{(i)}$: current/predicted probability

$$y_o^{(i)} = \frac{\left(y_o^{(i)}\right)^{1/T}}{\sum_j \left(y_o^{(j)}\right)^{1/T}}, \quad \hat{y}_o'^{(i)} = \frac{\left(\hat{y}_o^{(i)}\right)^{1/T}}{\sum_j \left(\hat{y}_o^{(j)}\right)^{1/T}}$$

Experiment Setting and Results

- **Settings**: The resnet101 is used as our base model. The task is first sequentially trained on the training set. The total epoch of whole dataset is about 12*2(for each task) in total.

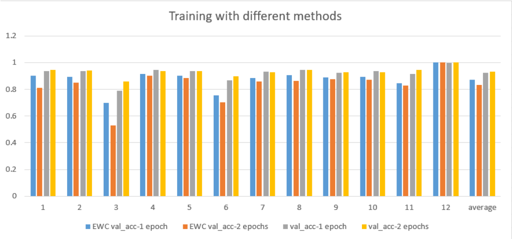


Figure 3: Training with different methods and configurations, X represents for task name and average accuracy, while y is the accuracy.

Conclusion

- We first training the task sequentially and got 93.33% average accuracy at Validation set across task 1 to task 12. However, during the training process, the accuracy test on Validation set is nearly 100%, which means the model is suffering from the catastrophic forgetting problem in sequentially training.
- EWC is then employed on the training process, however, the result is getting worse.
- Sequentially Training will be suffering from the catastrophic forgetting problem.
- Less training epochs out performances large training epochs.
- EWC training has a worse result due to the fact the estimation of Fisher Information Matrix might be biased estimated.
- In the future, we will focus on the dynamic graph for better preserving the memory of pervious task.

Discussions

- From our observations, the expandable network outperforms single model, but why?
- Can we use explainable models for better memorizing previous tasks? For example, by disentangling the environment information like illumination, occlusion, clutter, and perspectives w.r.t target object, as well as the observing distance between the camera and the target object.

References

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al.

Overcoming catastrophic forgetting in neural networks.

Proceedings of the national academy of sciences, 114(13):3521–3526, 2017.

Zhizhong Li and Derek Hoiem.

Learning without forgetting.

IEEE transactions on pattern analysis and machine intelligence, 40(12):2935–2947, 2017.

- Thank You.