How human actually learn? Continual learning: A more practical setting for image classification

Zhehao Zhang
Shanghai Jiao Tong University
zzh12138@sjtu.edu.cn

Abstract

Is traditional supervised learning always suitable for image classification? Do human beings learn things through supervised learning fashion? The answer to both above questions is absolutely no. As a result, in recent years, the setting of continual learning has drawn more and more attention. In my project, I study continual learning (especially class incremental learning), analyze its challenges, implement several classical methods. Besides, I **formalize a new setting** which is more practical and more similar to the process of human learners. After that, I **propose a new method** to solve this harder problem with performance better than classic methods. Furthermore, I also **learn new things** such as Fisher Information Matrix, Knowledge distillation and Causal inference.

1 Introduction

1.1 Background and settings

In the field of image classification, supervised learning has always been the mainstream technique since the era of deep learning. According to the traditional supervised learning fashion, the input dataset with all classes is shuffled and then fed into the classification model during training. During a fixed period of time, the probability to be encountered by the model is equal for every class. As a result, the model will update its parameters to minimize the loss function of every class equally without any bias (if there is no imbalance between classes in the dataset).

However, the learning process of the traditional supervised learning fashion is quite different from that of human learners. For example, it is not practical for a person to look through all classes of objects before classifying images. On the contrary, We always learn new things through our life and these new things are impossible to show up before a certain moment. This is why the setting of continual learning is also called Lifelong learning, and incremental learning.

There are different sub-settings in the field of continual learning, such as task incremental, domain incremental, and class incremental. The difference between each of them can be found in [1], and we will only discuss the setting of class incremental. The setting of class incremental learning is clearly illustrated by Figure 1.

1.2 Challenges and goals

Just like human beings, a classification model (eg. deep neural networks) will also witness the forgetting process. As a result, the major challenge is to learn without the so-called catastrophic forgetting: performance on a previously learned task or domain should not significantly degrade over time as new tasks or domains are added[3].

SJTU AI3604 Computer Vision 2021 Final Project.

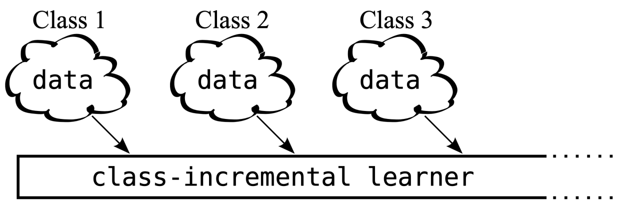


Figure 1: Class-incremental learning: an algorithm learns continuously from a sequential data stream in which new classes occur. At any time, the learner is able to perform multi-class classification for all classes observed so far. [2]

Why does forgetting happen? To systematically explain the (anti-)forgetting in class incremental learning in terms of the causal effect of the old knowledge, we frame the data, feature, and label in each incremental learning step into causal graphs [4]. The causal graph is a directed acyclic Bayesian graphical model, where the nodes denote the variables, and the directed edges denote the causality between two nodes.

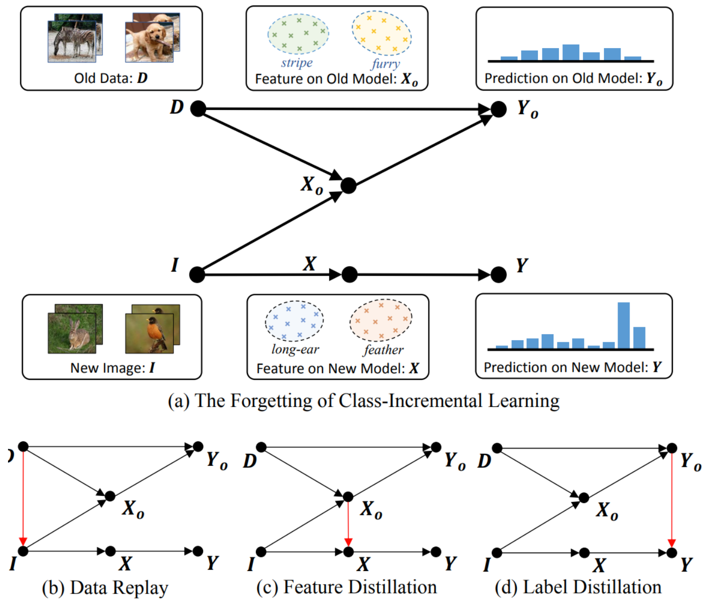


Figure 2: The proposed causal graphs explaining the forgetting and anti-forgetting in CIL. We illustrate the meaning of each node in the CIL framework in (a). The comparison between (a)-(d) shows the key to combating forgetting is the causal effect of old data.[5]

In the training process of class incremental learning, we consider the difference between the predictions Y with and without the existence of old data D. The effect of old data D on Y can be formulated as :

$$\begin{aligned} Effect_D &= P(Y=y|do(D)=d) - P(Y=y|do(D)=0) \\ &= P(Y=y|D=d) - P(Y=y|D=0) \end{aligned} \tag{1}$$

where $P(Y|do(D)) = P(Y|D)$ since D has no parents.
As observed in Figure 2 (a), all the causal paths from D to Y is blocked by colliders. For example, the path $D \to X_0 \leftarrow I \to X \to Y$ is blocked by the collider X_0. Applying the property of colliders in [6], we can get:

$$\begin{aligned} Effect_D &= P(Y=y|D=d) - P(Y=y|D=0) \\ &= P(Y=y) - P(Y=y) = 0 \end{aligned} \tag{2}$$

The zero effect means that D has no influences on Y in the new training – forgetting.

2 Related work

According to my survey, there are mainly three different families of methods in the field of class incremental learning based on how task-specific information is stored and used throughout the sequential learning process. These are replay methods, regularization-based methods, parameter isolation methods[3]. The reason for them to alleviate the forgetting can be easily understood by (b),(c),(d) in Figure 2 and in general, they break the colliders in different ways.

In the project, I mainly focus on the replay methods, regularization-based methods, and implement Learning without forgetting [7], Elastic Weight Consolid [8], Incremental Classifier and Representation [2]. In the following subsections, brief introductions of these methods will be discussed first and the experiment results will be presented later with analysis.

Besides implementing existing methods to conduct experiments in the setting of traditional continual learning settings, I also come up with a more practical setting in our daily life and optimize the existing methods to perform better.

2.1 Learning without forgetting [7](LwF)

The main idea of this paper is to make the output of the model trained on old task (or class) to be similar to that of the old model, which is illustrated in Figure3. It is achieved by using Knowledge Distillation loss, which was found by Hinton et al. [9] to work well for encouraging the outputs of one network to approximate the outputs of another. We denote the old model's output of the old task as y_o and new model's output of the old task as $\hat{y_o}$. The previous Knowledge Distillation loss can be defined as following.

$$L_{old}(y_o, \hat{y_o}) = -H(y'_o, \hat{y}'_o) = -\sum_{i=0}^{l} y'^{(i)}_o \log \hat{y_o}'^{(i)} \tag{3}$$

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

As a result, the total loss can be defined as following:

$$L_{total} = \lambda_o L_{old}(y_o, \hat{y_o}) + L_{new}(y_n, \hat{y_n}) + R$$

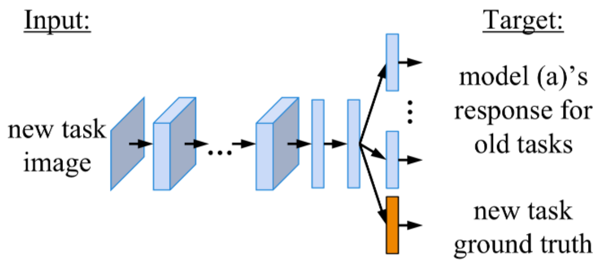


Figure 3: Learning target for LwF

2.2 Elastic Weight Consolidation(EWC)[8]

A deep neural network is composed of multiple layers of fully connected layers followed by element-wise activation functions (non-linear). The training process includes adjusting the set of weights and biases θ of the linear layers. Many configurations of θ will result in the same performance[10][11]. For this overparameterization, it is possible that there is a solution for task

B, θ_B^* , that is close to the previously found solution for task A, θ_A^*. In other words, the low error regions for task A and task B have some intersection as shown in Figure4.

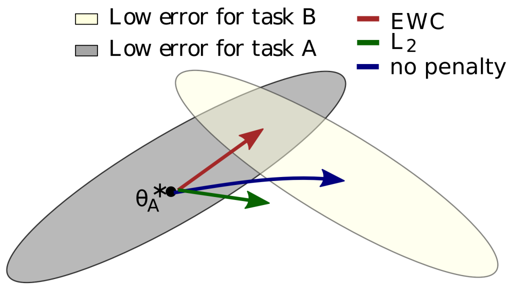


Figure 4: Three different ways' training trajectories are illustrated in a schematic parameter space.

As a result, there might be some parameters that are less useful and others are more valuable. In the l_2 constraint case, each parameter is treated equally. Here, we want to use the diagonal components in Fisher Information Matrix to identify which parameters are more important to task A and apply higher weights to them. Consequently, The loss function of EWC is

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

$L_B(\theta)$ represents the loss function of task B and F represents the Fisher Information Matrix.

2.3 Incremental Classifier and Representation Learning(iCaRL)[2]

The usage of exemplar rehearsal for class-IL was first proposed in Incremental Classifier and Representation Learning (iCaRL). Its method stores a subset of exemplars per class, selected to best approximate class means in the learned feature space. At test time, the class means are calculated for nearest-mean classification based on all exemplars[3]. The detailed algorithm can be found in the supplementary material due to the page limit.

3 Methodology

In the process of studying the class incremental learning, I gradually realize that although the setting of class incremental learning is more practical than traditional supervised learning, the data stream always comes from the same dataset. This condition is not always satisfied in real life. For example, it is impossible to learn knowledge that is illustrated in the same style in our life. For human beings, when learn to recognize dog, we are not able to guarantee that the dogs have the same style. For classification models, it should also have the ability to learn well from different datasets in the setting of class incremental learning.

As a result, I conduct the experiment for the above methods to learn from both MNIST and SVHN. To be specific, I feed the model with (0,1) in MNIST, (2,3) in SVHN, (4,5) in MNIST, (6,7) in SVHN, (8,9) in MNIST. This setting is illustrated in Figure 5.

As a result, I propose another algorithm to solve the impact of different datasets, which is illustrated in Algorithm 1. The main difference between my algorithm's first part of the total loss function from LwF loss is that my method expects the model to imitate the output of all models trained on the same dataset. On the contrary, LwF expects the model to imitate the output of the last model whatever it was trained on. The reason for doing this is that I observe that it is harmful to the model to imitate the output of past models trained on different datasets. On the one hand, the first part of total loss has the power for the model not to forget other tasks in the same dataset. On the other hand, the second part of total loss has the power for the model not to forget tasks in the other data. The third part has the power to learn news. These conclusions has been approved through my experiment(not presented due to page limit). The trade-off between the performance of the same dataset or different datasets can be changed through hyper-parameter λ_1 and λ_2.

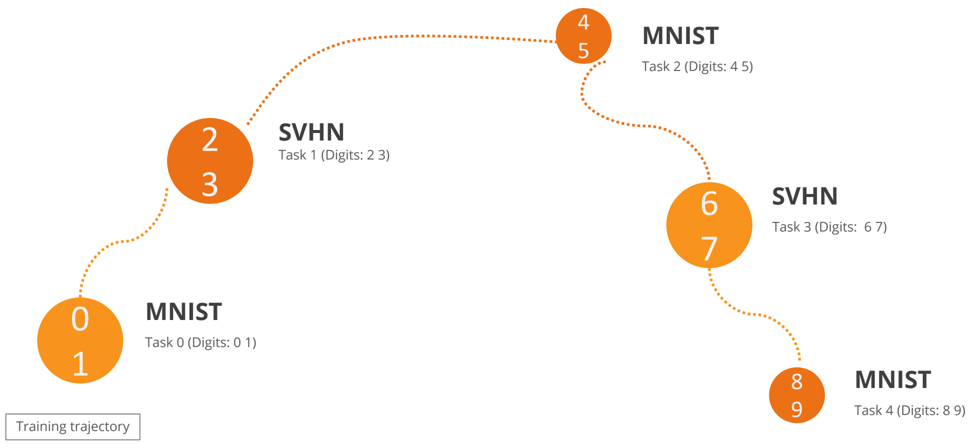


Figure 5: A more practical setting in class incremental learning with different datasets. Models need to have a better ability to generalize between different datasets.

Algorithm 1: Improvement of current method to work well for different dataset

```
Data: Stream data from MNIST and SVHN with image $x_i$ and label $Y_i$
Initialization: A classifier: $f$, model buffer : $B$. hyper parameter $\lambda_1$and$\lambda_2$
. for $t \leftarrow 0$ to number of tasks do
    if $t > 0$ then
        $Loss = 0$
        Take out the model $f_i$ out from model buffer $B$.
        Feed the image from the current task to $f_i$ to get the Pseudo label $Y_i'$.
        The total loss consists of three parts: distillation loss, Fisher information loss, new task
          cross entropy.
            $Loss = \sum_{i,d(i)=d(n)} \lambda_1 L_{old}(Y_i', \hat{Y}_i') + \sum_i \lambda_2 F_i(\theta_i - \theta^*_{old,i})^2 + L_{new}(Y_t, \hat{Y}_t)$
    else
        Train the model $f(x)$ with cross entropy loss function: $L_{new}(\hat{Y}_t, Y_t)$
    Store the current trained model in the model buffer $B$
    Calculate Fisher information.
```

4 Experimental results

4.1 Base experiment results

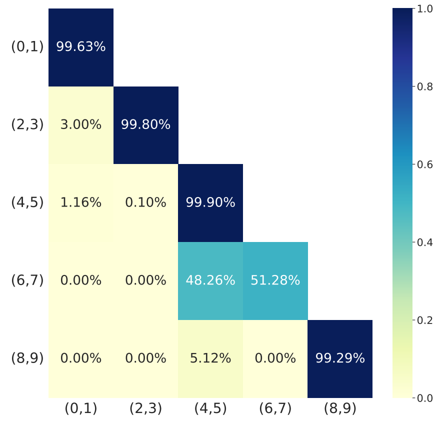
In the project, I conduct the class incremental learning experiments using the above methods. To better visualize the degree of overcoming catastrophic forgetting, I use the heat map to depict the accuracy after training each task for a single method. Apart from accuracy, I also adopt a special metric called Forgetting:

$$F_{ti} = \max_i Acc(i) - Acc(t)$$

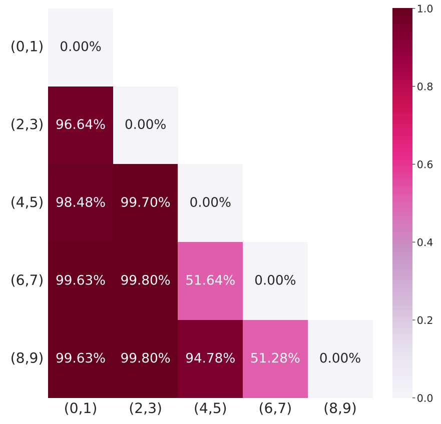
This metric indicates that after training task t, the accuracy drop of task i compared with its best performance.
The datasets used in the experiment are MNIST, SVHN, CIFAR100. Other details including hyper-parameters can be found in the supplementary.

4.2 Discussion on the experiment result

The result in Figure6 indicates that the naive finetuning method results in severe forgetting. Namely, finetuning always only performs well in the current task. This phenomenon can be summarized as task-recency bias, which is commonly observed in continual learning.

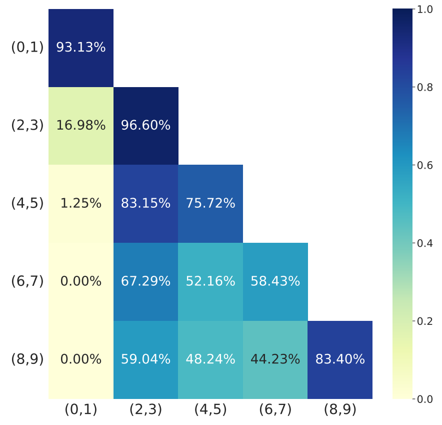


(a) MNIST accuracy in finetuning

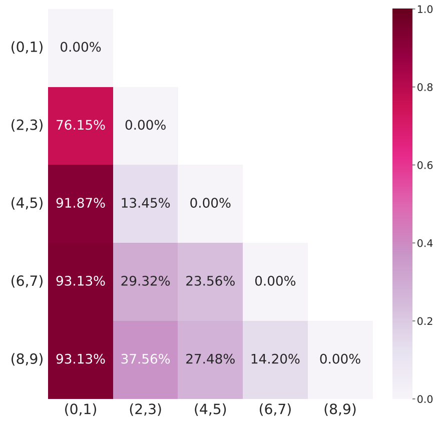


(b) MNIST forgetting in finetuning

Figure 6: Accuracy and forgetting in MNIST by finetuning. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) MNIST accuracy in LwF



(b) MNIST forgetting in LwF

Figure 7: Accuracy and forgetting in MNIST by LwF. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i

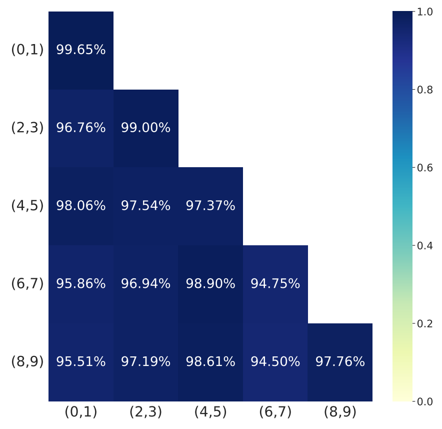
Table 1: Overall accuracy of each method after training each task on MNIST

Method	Task0	Task1	Task2	Task3	Task4
Finetuning	99.63	51.4	33.72	24.885	20.882
LwF	99.47	51.295	46.73	36.26	36.57
EWC	99.63	50.505	36.663	31.65	26.642
iCaRL	99.65	97.88	97.66	96.41	96.714

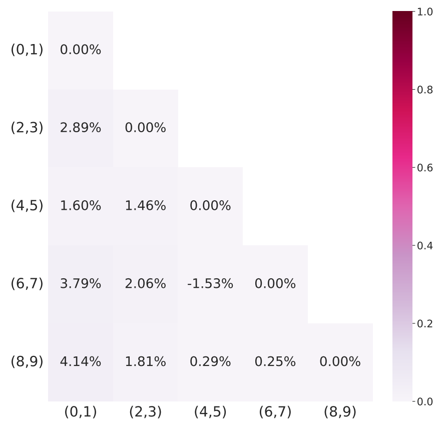
Table 2: Overall accuracy of each method after training each task on SVHN

Method	Task0	Task1	Task2	Task3	Task4
Finetuning	95.4	58.09	30.52	28.1125	22.044
LwF	93.13	56.79	53.37	44.47	46.982
EWC	94.93	49.52	39.16	18.2125	12.89
iCaRL	99.65	97.88	97.65	96.61	96.714

Compared with finetuning, the performance of LwF (can be found in Figure7) is much better. To be specific, this method can somehow alleviate forgetting of the past tasks.

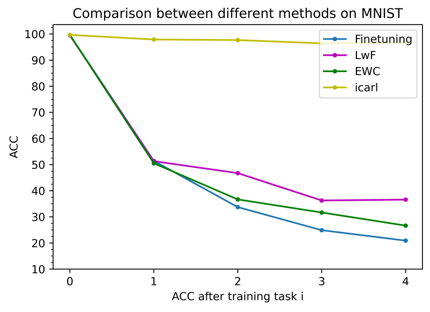


(a) MNIST accuracy in iCaRL

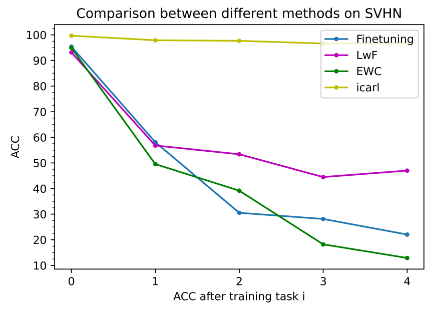


(b) MNIST forgetting in iCaRL

Figure 8: Accuracy and forgetting in MNIST by iCaRL. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) Different methods's accuracy on MNIST



(b) Different methods's accuracy on SVHN

Figure 9: Accuracy and forgetting in MNIST by iCaRL. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i

Table 3: Overall accuracy of each method after training each task on CIFAR100

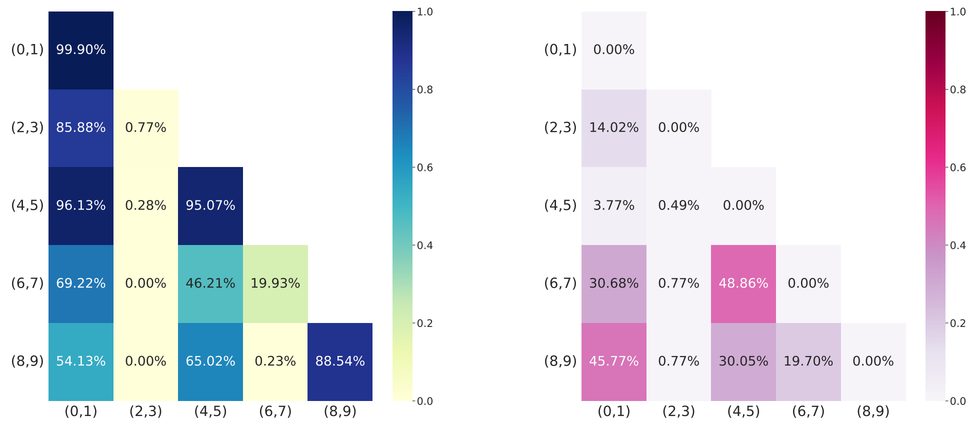
Method	Task0	Task1	Task2	Task3	Task4
Finetuning	60.50	37.15	25.9	23.15	16.55
LwF	54.15	37.45	29.05	25.4	22.62
EWC	59.15	41.875	34.183	29.8375	27.68
iCaRL	53.45	45.6	40.35	36.25	33.4

The heat map of other experiments can be found in the supplementary due to the page limit.

These results also indicate that replay-based methods such as iCaRL perform much better than regularization-based methods. This is quite intuitive and there is a rigorous analysis in [12] which tells us the regularization-based methods is NP-hard in the perspective of set theory and the replay-based methods are supposed to perform better.

4.3 Experiments with different dataset

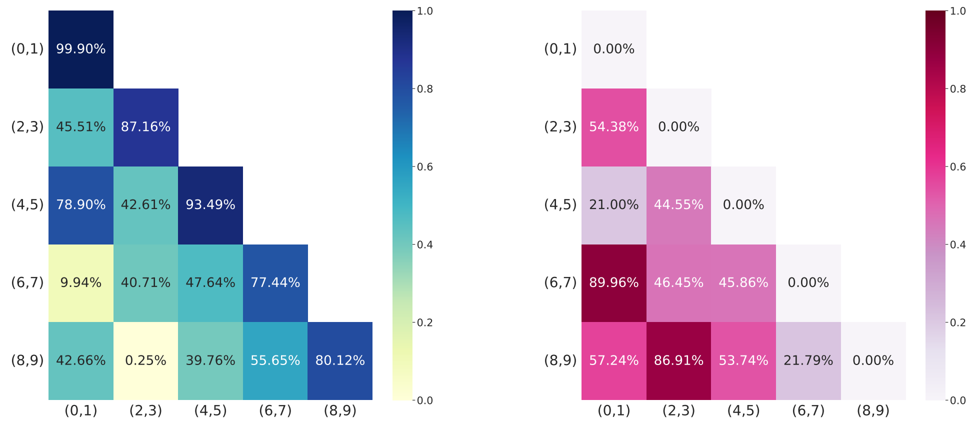
This part is for the new setting proposed in Part3. The experiment result of Algorithm 1 is illustrated in Figure11. The full experiment results can be found in Table 4 and my algorithm always has the best performance during the training process. Figure 10a shows that the above regularization-based methods can not perform well in this setting.



(a) MNIST and SVHN accuracy in LwF

(b) MNIST and SVHN forgetting in LwF

Figure 10: A more practical setting with MNIST and SVHN trained in turn: MNIST:(0,1),(4,5),(8,9) SVHN:(2,3),(6,7). Different dataset severely affect the performance.



(a) MNIST and SVHN accuracy in My algorithm

(b) MNIST and SVHN forgetting in My algorithm

Figure 11: My algorithm perform much better than LwF in the setting of class incremental learning with different dataset.

Table 4: Overall accuracy of each method after training each task on different dataset. In general, my method achieves the best performance in this more complicated setting

Method	Task0	Task1	Task2	Task3	Task4
Finetuning	99.9	55.82	49.8	28.01	21.07
LwF	99.9	43.325	63.82	33.84	41.584
EWC	99.9	59.35	38.29	31.34	25.81
My algorithm	**99.9**	**66.34**	**71.67**	**43.9**	**43.7**

5 Conclusion

In this project, I studied the setting of class incremental learning and implement several classic methods and a unified framework for class incremental learning. Besides, I propose a more practical setting base on it with different datasets. Because the classic methods can not work well in this setting, I propose a new algorithm to solve this more complicated problem. After conducting numerous experiments, My algorithm can beat these classic methods.

[1] Y.-C. Hsu, Y.-C. Liu **and** Z. Kira, "Re-evaluating continual learning scenarios: A categorization and case for strong baselines," *CoRR*, **jourvol** abs/1810.12488, 2018. arXiv: `1810.12488`. [Online]. Available: `http://arxiv.org/abs/1810.12488`.

[2] S.-A. Rebuffi, A. Kolesnikov, G. Sperl **and** C. H. Lampert, "Icarl: Incremental classifier and representation learning," **in** *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017.

[3] M. Delange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh **and** T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **pages** 1–1, 2021. DOI: `10.1109/TPAMI.2021.3057446`.

[4] J. Pearl, "Causality: Models, reasoning, and inference, second edition," *Causality*, **jourvol** 29, **january** 2000. DOI: `10.1017/CBO9780511803161`.

[5] X. Hu, K. Tang, C. Miao, X.-S. Hua **and** H. Zhang, "Distilling causal effect of data in class-incremental learning," **in** *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2021, **pages** 3957–3966.

[6] J. Pearl, M. Glymour **and** N. Jewell, *Causal Inference in Statistics: A Primer*. Wiley, 2016, ISBN: 9781119186854. [Online]. Available: `https://books.google.co.uk/books?id=IqCECwAAQBAJ`.

[7] Z. Li **and** D. Hoiem, "Learning without forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **jourvol** 40, **number** 12, **pages** 2935–2947, 2018. DOI: `10.1109/TPAMI.2017.2773081`.

[8] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran **and** R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, **jourvol** 114, **number** 13, **pages** 3521–3526, 2017, ISSN: 0027-8424. DOI: `10.1073/pnas.1611835114`. eprint: `https://www.pnas.org/content/114/13/3521.full.pdf`. [Online]. Available: `https://www.pnas.org/content/114/13/3521`.

[9] G. Hinton, O. Vinyals **and** J. Dean, *Distilling the knowledge in a neural network*, 2015. arXiv: `1503.02531 [stat.ML]`.

[10] Hecht-Nielsen, "Theory of the backpropagation neural network," **in** *International 1989 Joint Conference on Neural Networks*, 1989, 593–605 vol.1. DOI: `10.1109/IJCNN.1989.118638`.

[11] H. J. Sussmann, "Uniqueness of the weights for minimal feedforward nets with a given input-output map," *Neural Networks*, **jourvol** 5, **number** 4, **pages** 589–593, 1992, ISSN: 0893-6080. DOI: `https://doi.org/10.1016/S0893-6080(05)80037-1`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0893608005800371`.

[12] J. Knoblauch, H. Husain **and** T. Diethe, "Optimal continual learning has perfect memory and is NP-hard," **in** *Proceedings of the 37th International Conference on Machine Learning*, H. D. III **and** A. Singh, Eds., **jourser** Proceedings of Machine Learning Research, **volume** 119, PMLR, 13–18 Jul 2020, **pages** 5327–5337. [Online]. Available: `https://proceedings.mlr.press/v119/knoblauch20a.html`.