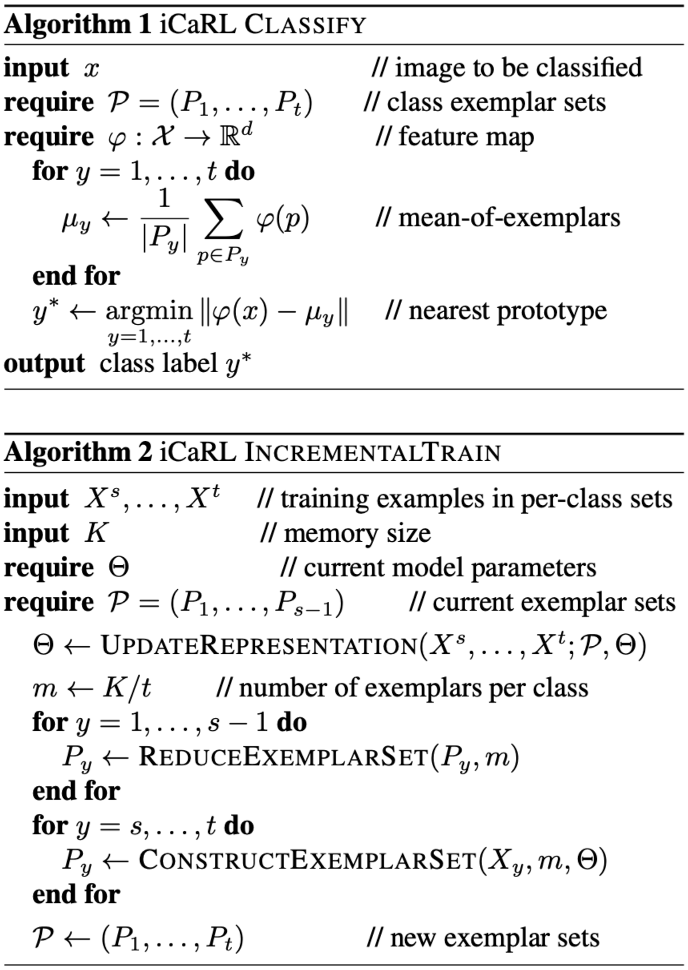
Supplementary Material for Final Project

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

1 Detailed algorithm of icarl



Algorithm 1 iCaRL CLASSIFY

```
input $x$                                   // image to be classified
require $\mathcal{P} = (P_1, \dots, P_t)$   // class exemplar sets
require $\varphi : \mathcal{X} \to \mathbb{R}^d$   // feature map
  for $y = 1, \dots, t$ do
    $\mu_y \leftarrow \frac{1}{|P_y|} \sum_{p \in P_y} \varphi(p)$   // mean-of-exemplars
  end for
  $y^* \leftarrow \operatorname{argmin}_{y=1,\dots,t} \|\varphi(x) - \mu_y\|$   // nearest prototype
output class label $y^*$
```

Algorithm 2 iCaRL INCREMENTALTRAIN

```
input $X^s, \dots, X^t$   // training examples in per-class sets
input $K$                 // memory size
require $\Theta$          // current model parameters
require $\mathcal{P} = (P_1, \dots, P_{s-1})$   // current exemplar sets
  $\Theta \leftarrow$ UPDATEREPRESENTATION($X^s, \dots, X^t; \mathcal{P}, \Theta$)
  $m \leftarrow K/t$   // number of exemplars per class
  for $y = 1, \dots, s-1$ do
    $P_y \leftarrow$ REDUCEEXEMPLARSET($P_y, m$)
  end for
  for $y = s, \dots, t$ do
    $P_y \leftarrow$ CONSTRUCTEXEMPLARSET($X_y, m, \Theta$)
  end for
  $\mathcal{P} \leftarrow (P_1, \dots, P_t)$   // new exemplar sets
```

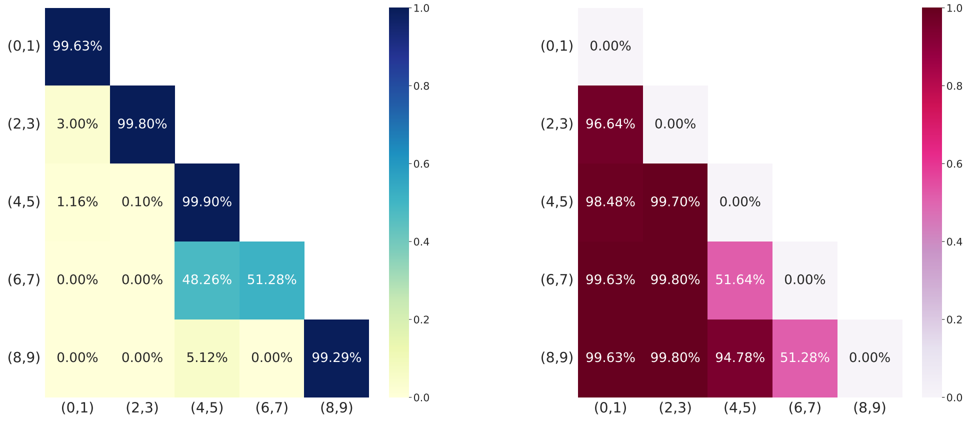
Figure 1: Algorithm used in iCaRL

SJTU AI3604 Computer Vision 2021 Final Project.

2 Experiment settings and full results

2.1 Base Experiment

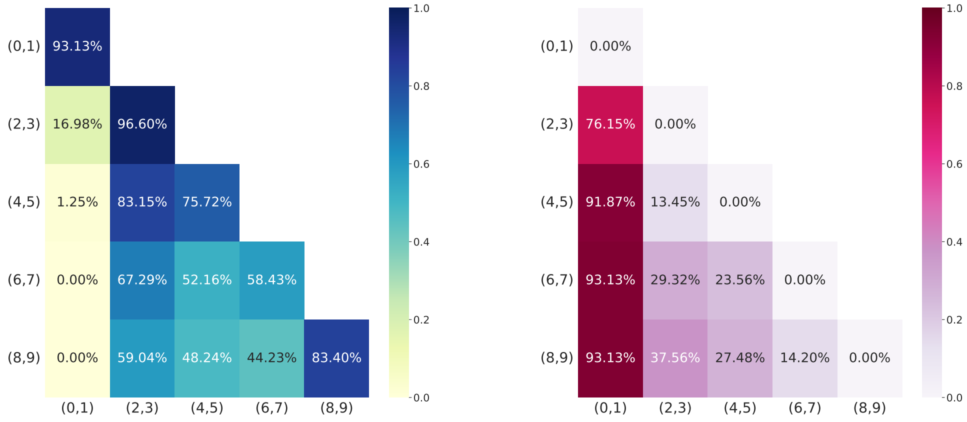
The corresponding part in the report is 2.4. I conduct experiment on MNIST, SVHN, CIFAR100 in the setting of class incremental learning. Epoch number is 5 for each task on on MNIST and SVHN. Epoch number is 50 for each task on on CIFAR100. The classifier is ResNet18. Learning rate is 0.1. In the training process, the learning rate is gradually decreasing and the decreasing factor is 3. The minimum learning rate is 0.0001. To prevent gradient exploding, I use clip_grad_norm and set the threhold value 10000. For Learning Without Forgetting (LwF), I set the hyper parameter $\lambda = 1$ and $T = 2$. For Elastic Weight Consolidation (EWC), I set set the hyper parameter $\lambda = 5000$ and $\alpha = 0.5$. For Incremental Classifier and Representation Learning (iCaRL), I set the number of exemplars 2000. GeForce RTX 2080Ti was used for computing. All experiments can be completed in a few minutes.



(a) MNIST accuracy in finetuning

(b) MNIST forgetting in finetuning

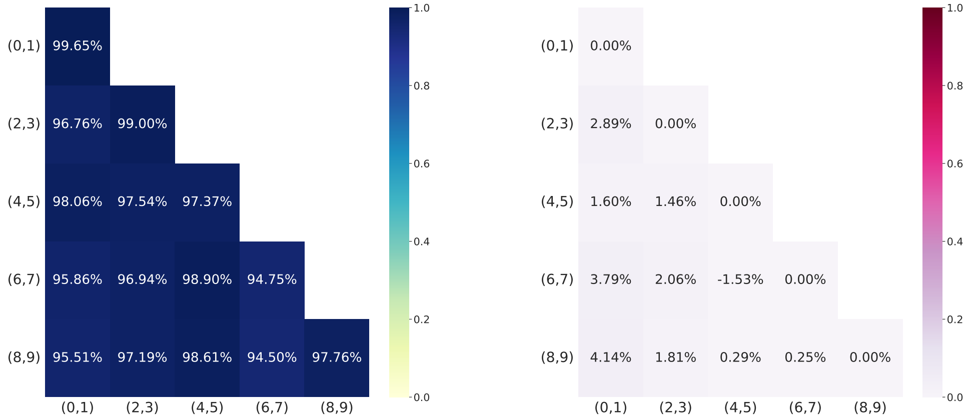
Figure 2: Accuracy and forgetting in MNIST by finetuning. Each element on the $i - th$ row and $j - th$ column represents the accuracy (or forgetting) of task j after training task i



(a) MNIST accuracy in LwF

(b) MNIST forgetting in LwF

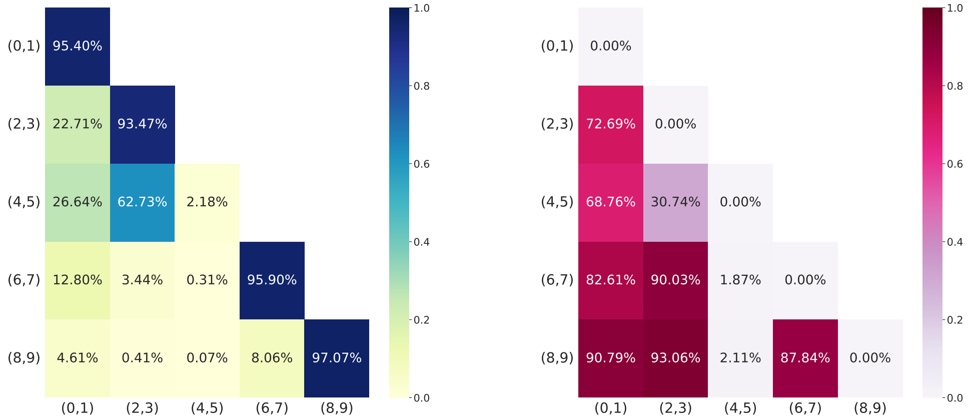
Figure 3: Accuracy and forgetting in MNIST by LwF. Each element on the $i - th$ row and $j - th$ column represents the accuracy (or forgetting) of task j after training task i



(a) MNIST accuracy in iCaRL

(b) MNIST forgetting in iCaRL

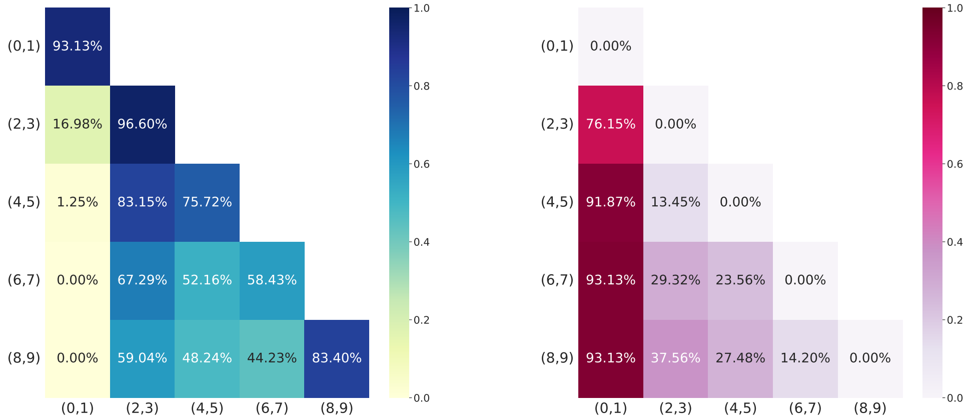
Figure 4: Accuracy and forgetting in MNIST by iCaRL. Each element on the $i - th$ row and $j - th$ column represents the accuracy (or forgetting) of task j after training task i



(a) SVHN accuracy in finetuning

(b) SVHN forgetting in finetuning

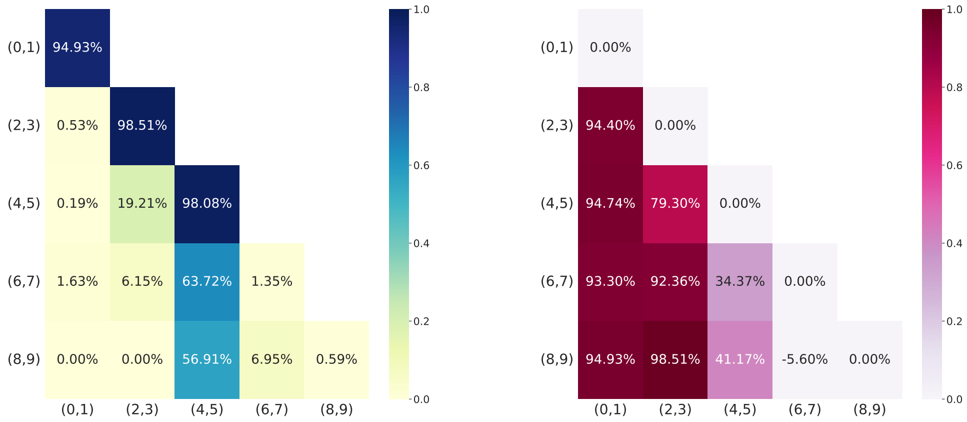
Figure 5: Accuracy and forgetting in SVHN by finetuning. Each element on the $i - th$ row and $j - th$ column represents the accuracy (or forgetting) of task j after training task i



(a) SVHN accuracy in lwf

(b) SVHN forgetting in lwf

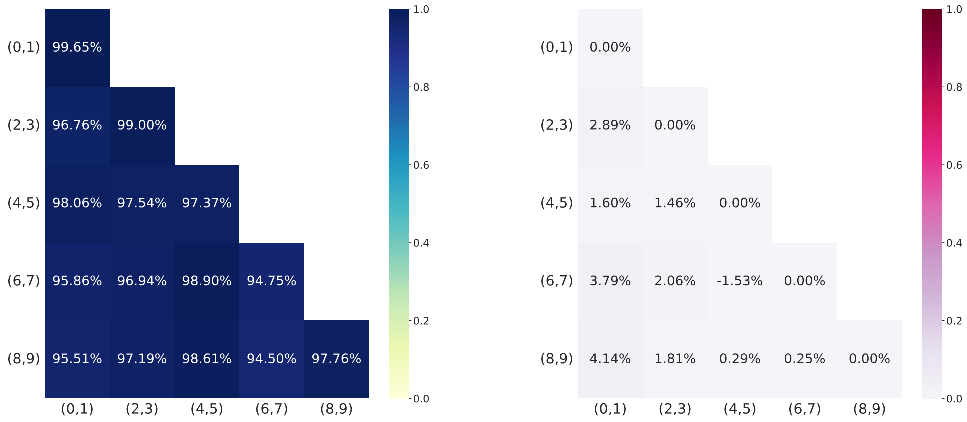
Figure 6: Accuracy and forgetting in SVHN by lwf. Each element on the $i - th$ row and $j - th$ column represents the accuracy (or forgetting) of task j after training task i



(a) SVHN accuracy in ewc

(b) SVHN forgetting in ewc

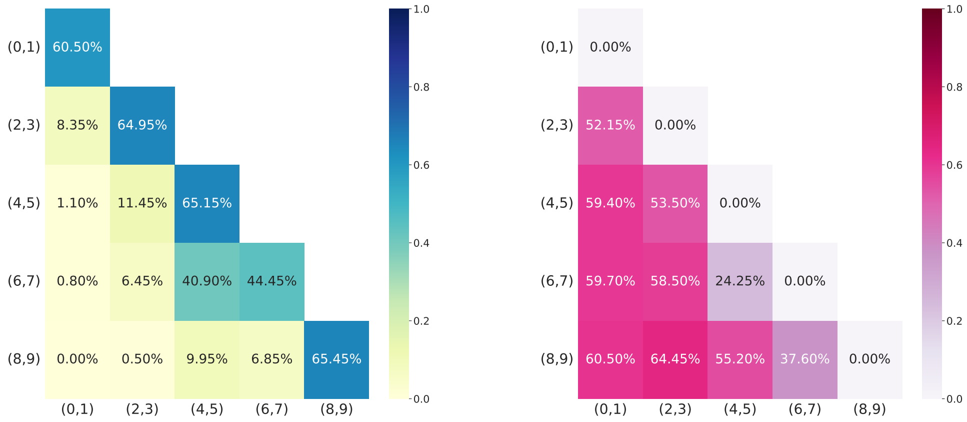
Figure 7: Accuracy and forgetting in SVHN by ewc. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) SVHN accuracy in icarl

(b) SVHN forgetting in icarl

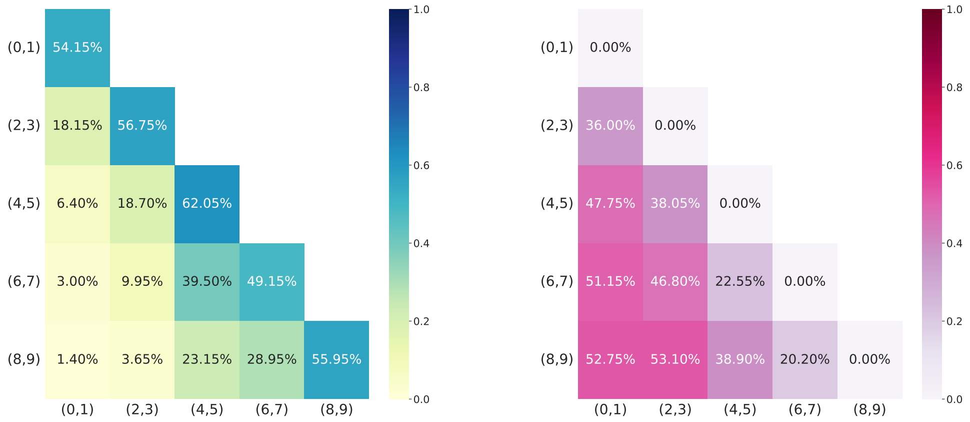
Figure 8: Accuracy and forgetting in SVHN by icarl. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) cifar100 accuracy in finetuning

(b) SVHN forgetting in icarl

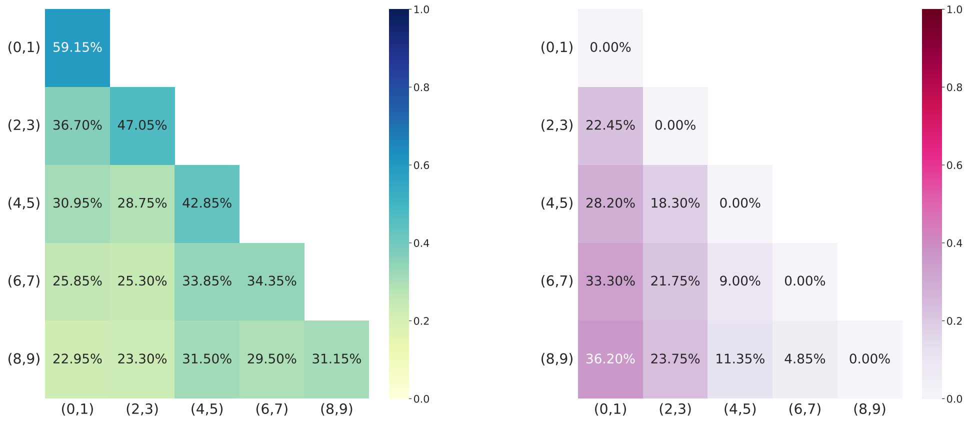
Figure 9: Accuracy and forgetting in cifar100 by finetuning. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) cifar100 accuracy in lwf

(b) SVHN forgetting in lwf

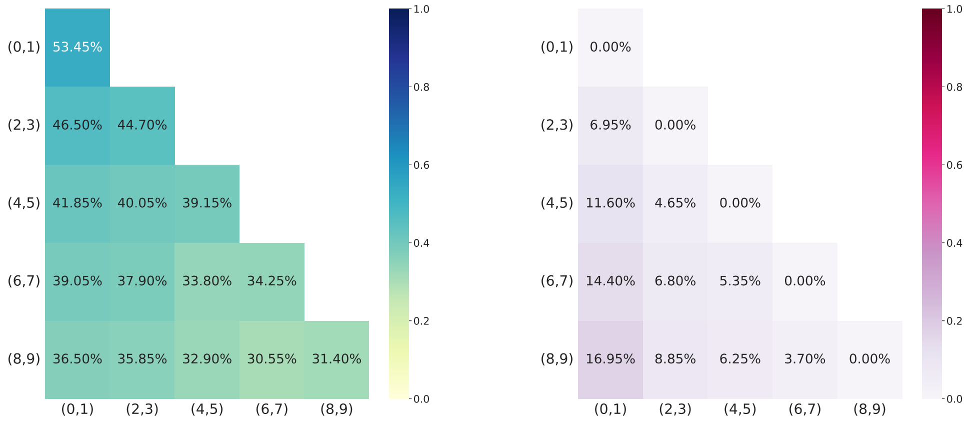
Figure 10: Accuracy and forgetting in cifar100 by lwf. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) cifar100 accuracy in ewc

(b) SVHN forgetting in ewc

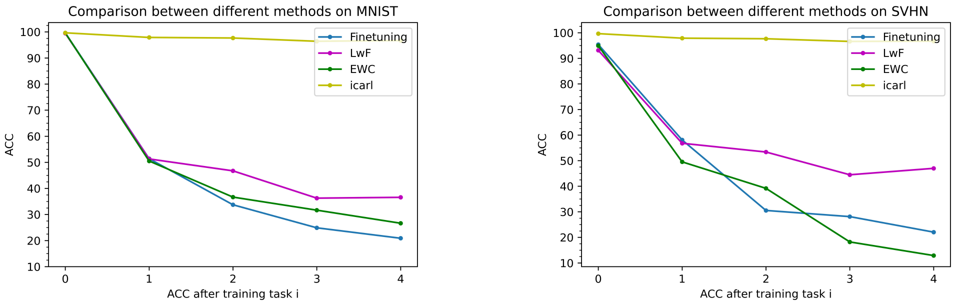
Figure 11: Accuracy and forgetting in cifar100 by ewc. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) cifar100 accuracy in icarl

(b) SVHN forgetting in icarl

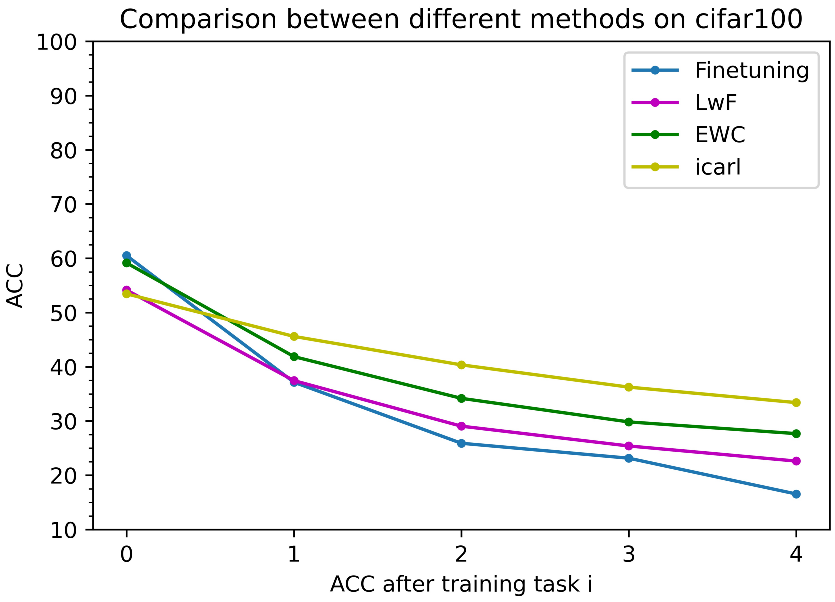
Figure 12: Accuracy and forgetting in cifar100 by icarl. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) Different methods's accuracy on MNIST

(b) Different methods's accuracy on SVHN

Figure 13: 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



.5

Figure 14: Different methods's accuracy on cifar100

2.2 Experiment results on a more practical setting

I propose a more practical setting in class incremental learning that is illustrated in Figure 15. 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. My goal is to let my model perform well on both datasets. In this experoment, I use the Resnet32 as the base model for every method. I set the epoch as 10 for every method. Learning rate is 0.1 and it will gradually decrease with the factor 3 during the training process. According to

$$Loss = \sum_{i,d(i)=d(n)} \lambda_1 L_{old}(Y_i', \hat{Y}_i') + \sum_{i} \lambda_2 F_i(\theta_i - \theta^*_{old,i})^2 + L_{new}(Y_t, \hat{Y}_t)$$

when $\lambda_1 = 1$ and $\lambda_2 = 5000$, the overall performance is the best. The parameter T in distillation loss is 2. $\alpha = 0.5$ in Fisher matrix.

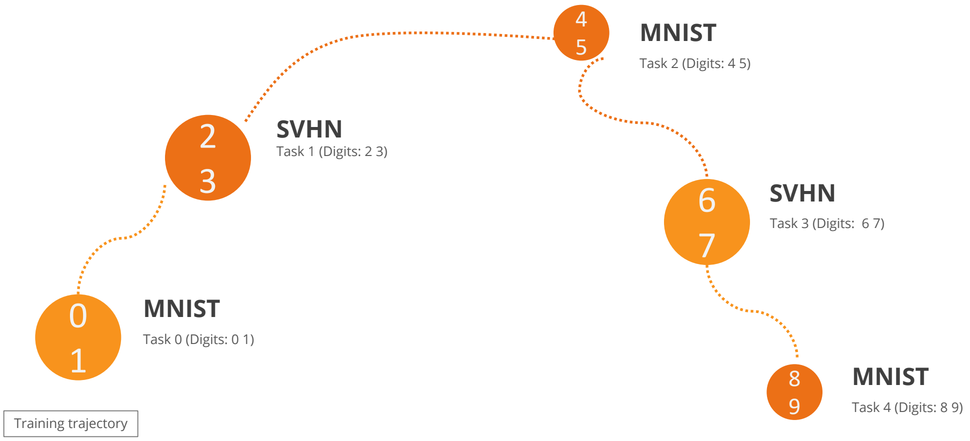
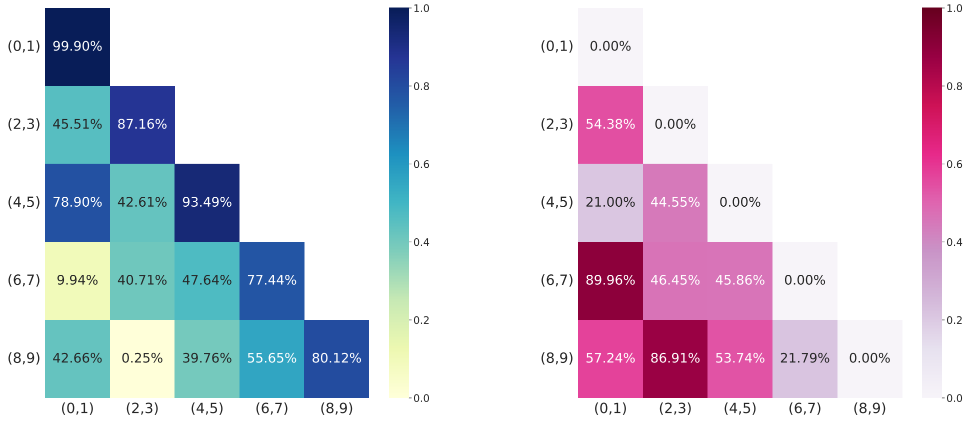


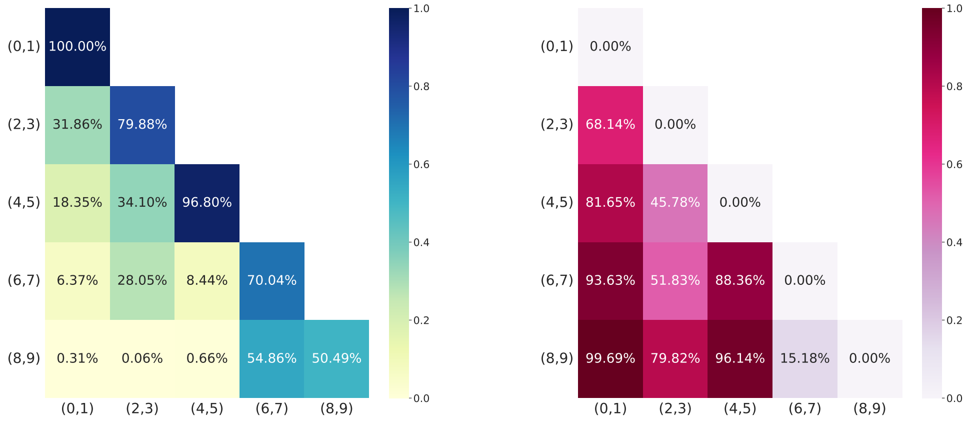
Figure 15: A more practical setting in class incremental learning with different datasets. Models need to have a better ability to generalize between different datasets.



(a) MNIST-SVHN accuracy in my algorithm

(b) MNIST-SVHN forgetting in my algorithm

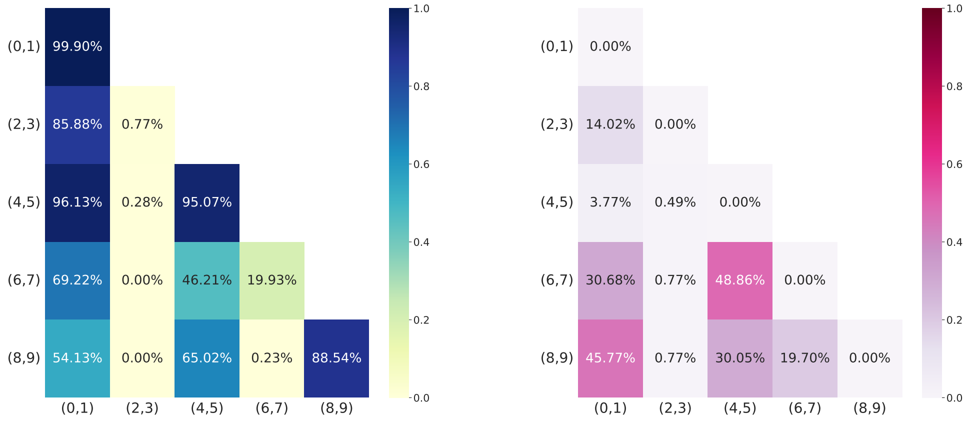
Figure 16: Accuracy and forgetting in MNIST-SVHN by my algorithm. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) MNIST-SVHN accuracy in finetuning

(b) MNIST-SVHN forgetting in ewc

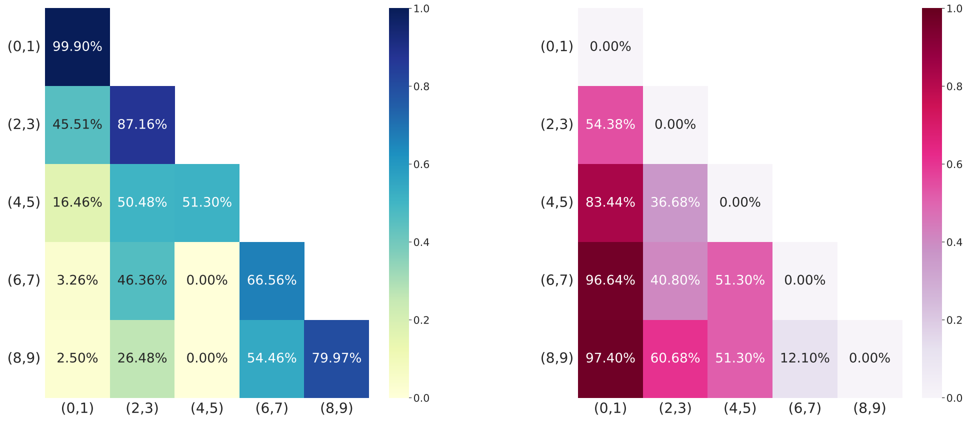
Figure 17: Accuracy and forgetting in MNIST-SVHN by finetuning. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i



(a) MNIST-SVHN accuracy in lwf

(b) MNIST-SVHN forgetting in lwf

Figure 18: Accuracy and forgetting in MNIST-SVHN 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



(a) MNIST-SVHN accuracy in ewc

(b) MNIST-SVHN forgetting in ewc

Figure 19: Accuracy and forgetting in MNIST-SVHN by ewc. Each element on the $i-th$ row and $j-th$ column represents the accuracy (or forgetting) of task j after training task i

3 Learned things

3.1 Fisher Information Matrix

To illustrate the meaning of Fisher information matrix, let us start by a example.Suppose we have a model parameterized by parameter vector θ that models a distribution $p(x|\theta)$. We want to maximize the likelihood wrt. θ. We define a score function:

$$s(\theta) = \nabla_\theta \log p(x|\theta)$$

which is the gradient of log likelihood function. We can further prove $\mathbb{E}_{p(x|\theta)}[\nabla \log p(x|\theta)] = 0$ easily.

We can define an uncertainty measure around the expected estimate. That is, we look at the covariance of score of our model. Taking the result from above:

$$\mathbb{E}_{p(x|\theta)}[(s(\theta) - 0)(s(\theta) - 0)^T] = \mathbb{E}_{p(x|\theta)}[\nabla_\theta \log p(x|\theta) \nabla_\theta \log p(x|\theta)^T]$$

We can then see it as information. The covariance of the score function above is the definition of Fisher Information. As we assume θ is a vector, the Fisher Information is in a matrix form, called Fisher Information Matrix.

The Fisher Information Matrix describes the covariance of the gradient of the log-likelihood function. Note that we call it "information" because the Fisher information measures how much the parameters tell us about the data.

Fisher Information Matrix has three key properties[1]: (i) It is equivalent to the second derivative of the loss near a minimum, (ii) it can be computed from first-order derivatives alone and is thus easy to calculate even for large models, and (iii) it is guaranteed to be positive semi-definite.

3.2 Knowledge distillation

It is well known that the great success of deep learning is mainly due to its scalability to encode large-scale data and to maneuver billions of model parameters. However, the resources in some devices ,eg.,mobile phones and embedded devices is limited for not only high computational complexity but also the large storage requirement. There is a small network that worked as a student network and a big network (or many networks) that worked as a teacher network. The teacher network was already trained on the entire dataset and performs well. The goal is to make the student network perform as well as the teacher network. During training, the loss function of the student network can be described as equation 3 which means the output logits of student networks should be similar to that of the teacher. The hyper-parameter T can make the output of the teacher network more smooth to make the student network learn with more ease and learn the hidden relationship between categories.

Besides directly imitating the output of teacher networks which is called response-based knowledge in **2021** , other knowledge including feature-based knowledge and relation-based knowledge.

In this project, I read several surveys on knowledge distillation and apply them to LwF. After that, I learn different algorithms in many papers in this area.

3.3 Causal inference

Causal inference has been a popular topic in the field of deep learning in recent years. We study causation because we need to make sense of data, to guide actions and policies, and to learn from our success and failures[2]. In the process of doing my project, I read the book Causal Inference in Statistics: A Primer [2]. The tool of causal inference indeed helps better understand how catastrophic forgetting happens and inspire me to find methods to alleviate it.

In order to deal rigorously with questions of causality, we must have a way of formally setting down our assumptions about the causal story behind a data set. To do so, This introduces the concept of the structural causal model, or SCM, which is a way of describing the relevant features of the world and how they interact with each other. They are directed acyclic graphs (DAGs)
A structural causal model M consists of

- A set of independent (exogenous) random variables $u = \{u_1, ..., u_n\} \in U$ with distribution $P(u)$.

- Functions $F = \{f_1, ..., f_n\}$
- Variable $X = \{X_1, ..., X_n\} \in V$ such that $X_i = f_i(PA_i, u_i)$

The variables in U are called exogenous variables, meaning, roughly, that they are external to the model; we choose, for whatever reason, not to explain how they are caused. The variables in V are endogenous. Every endogenous variable in a model is a descendant of at least one exogenous variable. Exogenous variables cannot be descendants of any other variables, and in particular, cannot be a descendant of an endogenous variable; they have no ancestors and are represented as root nodes in graphs. If we know the value of every exogenous variable, then using the functions in f , we can determine with perfect certainty the value of every endogenous variable. As a result, the prior distribution $P(u)$ and functions determine the distribution P^M.

The difference between intervening on a variable and conditioning on that variable should be obvious. When we intervene on a variable in a model, we fix its value. We change the system, and the values of other variables often change as a result. When we condition on a variable, we change nothing; we merely narrow our focus to the subset of cases in which the variable takes the value we are interested in. What changes, then, is our perception about the world, not the world itself.

- Intervention $I = do(X_i := \tilde{f}_i(\tilde{PA}_i, u_i))$
- We can simply write $do(X_i = x)$ to denote the intervention.

Using SCM, we can model a large number of problems. The book consists of several classic models such as Chains & Forks, Colliders, and so on which contain their unique properties.

4 Advice to next years CV students and instructors

For next years CV students, I advise that you should start early for this final project. I started quite early and was able to read lots of materials and conduct many experiments. A large amount of time makes me capable of coming up with my own ideas and make experiments to realize them. Besides, I advise that you should make more exploration. Instead of just finishing the task, more exploration can give you unexpected 'gifts' and learn more interesting things.
For the instructors, I advise that maybe you can decrease a little bit workload for this lecture. Both the homework and final project take a large amount of time. Besides, quizzes and the exam also make me quite anxious. After finishing the final project, I have to review the lectures to prepare for final exam. Another piece of advice is that I hope the instructor could spend more time on SOTA methods based on deep learning.

5 Reference

During the process of coding, I refer to the code of papers: Learning without forgetting, Overcoming catastrophic forgetting in neural networks, Icarl: Incremental classifier and representation learning, Online continual learning in image classification: An empirical survey, Class-incremental learning: survey and performance evaluation, A continual learning survey: Defying forgetting in classification tasks. These can be found on the github.

[1] R. Pascanu **and** Y. Bengio, *Revisiting natural gradient for deep networks*, 2014. arXiv: 1301.3584 [cs.LG].

[2] 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.