Class-Incremental Learning Experiments with the FruitNet Dataset

Oktay Kurt
University of Milan
kurt.oktay@outlook.com

***Abstract*—Class-Incremental Learning (Class-IL) is a branch of continual learning focused on sequentially adding new classes (tasks) into a single classification model. The critical challenge in Class-IL is *catastrophic forgetting*—the model often loses performance on previously learned classes when trained on new ones. This dilemma between retaining old knowledge (*stability*) and acquiring new knowledge (*plasticity*) is referred to as the *stability–plasticity dilemma*.**

This report presents an extensive empirical study using five approaches on an Indian fruits dataset called *FruitNet*:

- **Fine-Tuning – a naïve incremental approach (train only on new data each step),**
- **Joint – re-trains on all data at each task for an upper bound measure,**
- **iCaRL – incremental classifier and replay (stores exemplars),**
- **EWC – regularization-based (elastic weight consolidation),**
- **LwF – knowledge distillation (learning without forgetting).**

The experiments gauge:

- **How effectively each method preserves old classes' performance,**
- **How well each learns the new classes,**
- **The computational (time) and memory (model size) overhead.**

***Index Terms*—class-incremental learning, continual learning, catastrophic forgetting, FruitNet, iCaRL, EWC, LwF.**

I. Introduction

Class-Incremental Learning (Class-IL) is a subfield of continual learning that requires a model to learn new classes (tasks) sequentially while retaining performance on previously learned classes [1], [5], [7]. The main difficulty is *catastrophic forgetting*, where adapting to new data causes performance degradation on old tasks. This scenario poses the *stability–plasticity dilemma*: balancing the retention of past knowledge (*stability*) vs. effective integration of new information (*plasticity*).

Experiments were run on the *FruitNet* dataset [6] to compare the following:

- **Fine-Tuning**: trains only on data for the current task,
- **iCaRL**: a replay-based approach using exemplars [7],
- **EWC**: Fisher-based regularization [3],
- **LwF**: knowledge distillation from the old model [5].

A non-continual reference model, denoted **Joint**, is also included as an upper bound baseline.

II. Dataset: FruitNet

FruitNet [6] includes images of Indian fruits in categories *good*, *bad*, and *mixed*. Only the *good* and *bad* subsets are used for five fruit types (Apple, Banana, Guava, Lime, Orange). Each fruit has two classes (e.g., "Apple_Good" and "Apple_Bad").

All images are rescaled to 192×192 pixels and normalized. A single-head classification approach is applied, expanding the final layer by two outputs whenever a new fruit is introduced. Five tasks are defined, each adding two new classes (Apple, Banana, Guava, Lime, Orange). An 80%/20% train–test split is used.

The number of images for each label is nearly identical, so no additional balancing was performed. The approximate counts per label are:

- Apple_Bad: 1141
- Apple_Good: 1149
- Banana_Bad: 1087
- Banana_Good: 1113
- Guava_Bad: 1129
- Guava_Good: 1152
- Lime_Bad: 1085
- Lime_Good: 1094
- Orange_Bad: 1159
- Orange_Good: 1216

Each task introduces two classes (Bad/Good for each fruit), with the following total training and test samples:

- Task 0: 1831 training, 459 test
- Task 1: 1759 training, 441 test
- Task 2: 1824 training, 457 test
- Task 3: 1743 training, 436 test
- Task 4: 1899 training, 476 test

Label distributions per task are:

- Task 0 train: {0: 919, 1: 912}, test: {0: 230, 1: 229}
- Task 1 train: {2: 890, 3: 869}, test: {2: 223, 3: 218}
- Task 2 train: {4: 921, 5: 903}, test: {4: 231, 5: 226}
- Task 3 train: {6: 875, 7: 868}, test: {6: 219, 7: 217}
- Task 4 train: {8: 972, 9: 927}, test: {8: 244, 9: 232}

III. Methodology

A. Model and Training Setup

- **Network:** ResNet-18, pretrained on ImageNet.

Fig. 1: Sample images from the dataset classes.

- **Optimizer:** SGD with learning rate 10^{-4}, momentum 0.9, weight decay 10^{-4}.
- **Epochs per Task:** 5.
- **Batch Size:** 128.

B. Approaches and Baseline

Fine-Tuning: Trains only on the current task's data, making it prone to catastrophic forgetting.

iCaRL [7]: Maintains a replay buffer of exemplars from previous tasks.

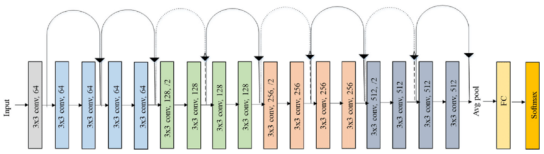

Fig. 2: ResNet-18 architecture.

EWC [3]: Uses a Fisher-based penalty to keep important weights close to their previous values.

LwF [5]: Uses knowledge distillation from previous models to mitigate forgetting.

Joint: Retrains on all accumulated data at each task step, serving as an upper bound reference rather than a continual method.

C. Performance Metrics

A 5×5 matrix R is tracked, where $R[t,k]$ is the accuracy on Task k after training on Task t. Key quantities include:

- **Current-Task Accuracy:** $R[t,t]$.
- **Past-Task Accuracy:** average of $R[t,0..t-1]$.
- **Forgetting:** $R[j,j]-R[4,j]$ for $j=0,\ldots,3$.
- **Plasticity:** average diagonal $\frac{1}{5}\sum_{t=0}^{4} R[t,t]$.
- **Stability:** negative sum of all forgetting for tasks 0 to 3.

IV. Experimental Results

A. Results Matrices (R)

Accuracy matrices R for each method are shown below.

Fine-Tuning

TABLE I: Fine-Tuning: R (5×5)

Train Step	Task0	Task1	Task2	Task3	Task4
0	0.95	0.04	0.17	0.05	0.25
1	0.23	0.91	0.03	0.03	0.36
2	0.10	0.49	0.97	0.11	0.00
3	0.04	0.36	0.48	0.94	0.03
4	0.02	0.20	0.41	0.43	0.89

Joint

TABLE II: Joint: R (5×5), Non-Continual Reference

Train Step	Task0	Task1	Task2	Task3	Task4
0	0.96	0.26	0.00	0.06	0.06
1	0.98	0.98	0.04	0.01	0.22
2	0.98	1.00	0.95	0.00	0.03
3	0.98	1.00	0.96	0.97	0.00
4	0.97	1.00	0.96	0.98	0.88

iCaRL
EWC
LwF

TABLE III: iCaRL: R (5×5)

Train Step	Task0	Task1	Task2	Task3	Task4
0	0.92	0.17	0.06	0.09	0.04
1	0.52	0.97	0.06	0.08	0.08
2	0.43	0.73	0.93	0.31	0.08
3	0.34	0.73	0.48	0.93	0.02
4	0.37	0.76	0.54	0.45	0.89

TABLE IV: EWC: R (5×5)

Train Step	Task0	Task1	Task2	Task3	Task4
0	0.94	0.47	0.09	0.02	0.03
1	0.48	0.91	0.10	0.00	0.27
2	0.44	0.78	0.89	0.15	0.40
3	0.24	0.27	0.20	0.98	0.11
4	0.19	0.44	0.34	0.73	0.62

TABLE V: LwF: R (5×5)

Train Step	Task0	Task1	Task2	Task3	Task4
0	0.78	0.05	0.01	0.14	0.00
1	0.32	0.56	0.15	0.12	0.14
2	0.14	0.10	0.80	0.24	0.07
3	0.01	0.04	0.07	0.92	0.25
4	0.00	0.02	0.02	0.20	0.80

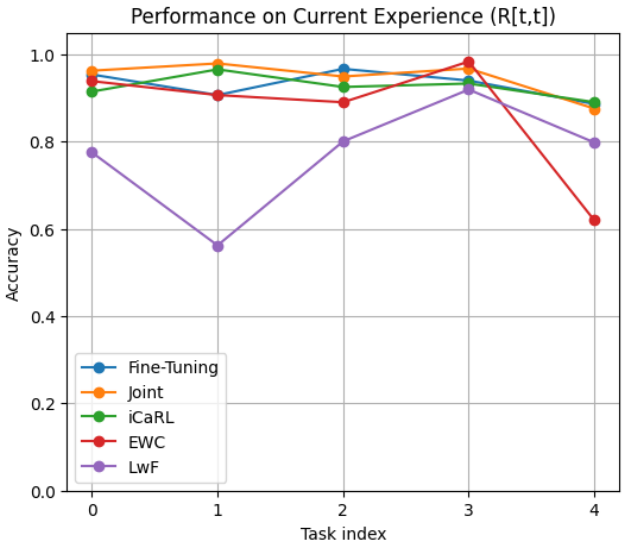

Fig. 3: Performance on current experience ($R[t,t]$).

B. Comparison Plots and Interpretations

Fine-Tuning, iCaRL, and EWC typically maintain high accuracy on the current task. LwF lags behind in some tasks. The Joint line is shown only for reference.

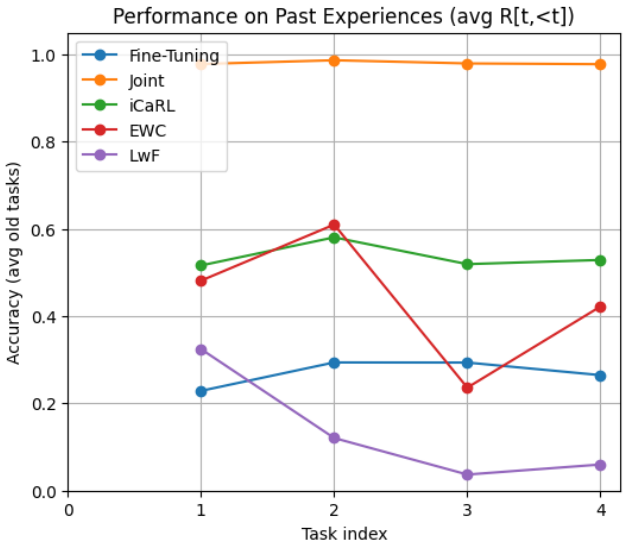

Fig. 4: Performance on past experiences (avg $R[t,<t]$).

iCaRL exhibits a moderate level of retention for previous tasks. EWC and Fine-Tuning degrade more as tasks progress, and LwF shows relatively weaker retention. The Joint curve remains near-perfect but is not a continual method.

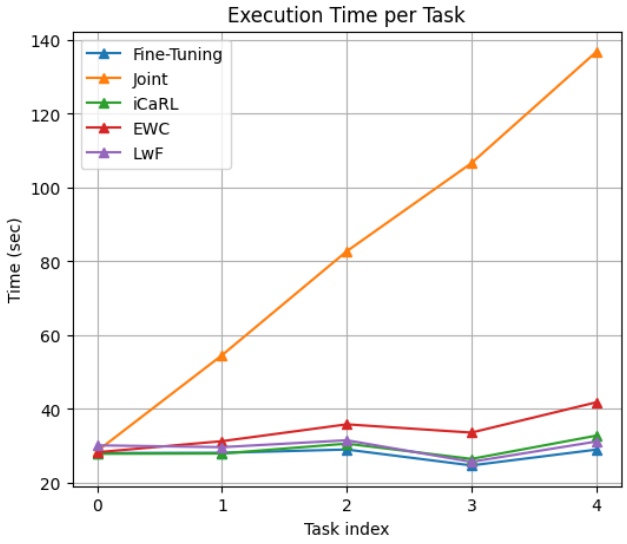

Fig. 5: Execution time per task.

Retraining on all data each step (Joint) increases time notably. Fine-Tuning, iCaRL, EWC, and LwF operate in a lower time range.

iCaRL finds a favorable balance between plasticity and stability. EWC is moderately effective, while Fine-Tuning

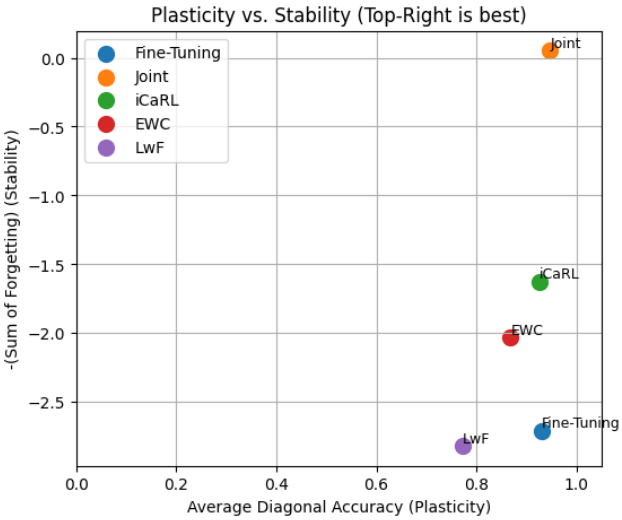

Fig. 6: Plasticity vs. stability (top-right is best).

has good plasticity but poor stability. LwF appears weaker in this setup. Joint is included as a reference.

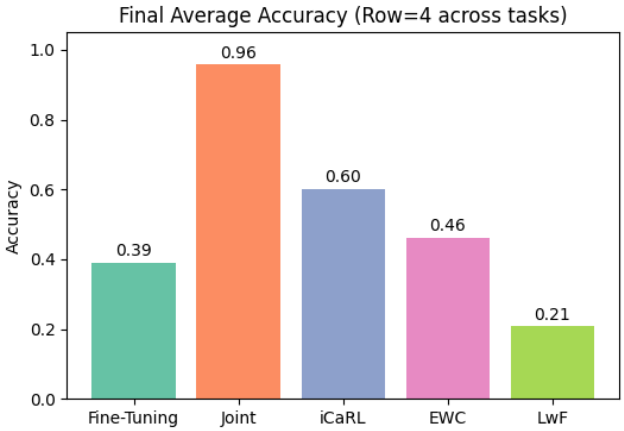

Fig. 7: Final Average Accuracy (Row=4 across tasks).

Figure 7 shows the final average accuracy (Row=4) for each method. Fine-Tuning obtains 0.39, Joint 0.96, iCaRL 0.60, EWC 0.46, and LwF 0.21. The Joint model benefits from retraining on all data, while iCaRL emerges as the top performer among the purely continual methods.

C. Total Training Time

Table VI compares total seconds from Task 0 to Task 4. Joint is strictly for reference.

V. Hyperparameter Search

A. iCaRL

Hyperparameter Selection for iCaRL: Although using 50 exemplars per class yields the highest final average accuracy and the lowest sum of forgetting, the performance gain over using 25 exemplars is marginal. Since doubling the memory requirement from 25 to 50 exemplars significantly increases memory usage without a significant improvement, selected 25 exemplars per class as a more practical trade-off.

TABLE VI: Total training time (all tasks)

Method	Total Time (s)
Fine-Tuning	138.9
Joint	409.3
iCaRL	145.7
EWC	170.9
LwF	148.2

TABLE VII: iCaRL hyperparameters

memory_per_class	Final Avg Acc	Sum of Forgetting
10	0.3969	2.35
25 (chosen)	0.5596	1.27
50	0.6672	0.92

B. EWC

TABLE VIII: EWC hyperparameters

ewc_lambda	Final Avg Acc	Sum of Forgetting
1	0.4367	1.72
10	0.3338	2.20
100 (chosen)	0.5122	1.46
1000	0.1002	1.31
10000	0.1002	0.30

Hyperparameter Selection for EWC: Observed that increasing the ewc_lambda parameter beyond 100 results in a dramatic reduction in the final average accuracy, with only a minimal decrease in the sum of forgetting. Therefore, a value of 100 for ewc_lambda was chosen as it provides the best balance between performance and forgetting mitigation.

C. LwF

Hyperparameter Selection for LwF: Although various combinations of distill_lambda and distill_temp were tested, the overall performance of LwF was insufficient compared to other methods. Among the tested combinations, the one with distill_lambda = 1.0 and distill_temp = 10 achieved the highest final average accuracy and was thus selected, despite its overall weaker performance.

VI. Discussion and Conclusions

Key Observations (Continual Methods):

- iCaRL successfully preserves old knowledge while maintaining strong results on new tasks.
- EWC helps mitigate forgetting but is less effective than iCaRL under these settings.
- LwF is weaker across both plasticity and stability measures in these experiments.

TABLE IX: LwF hyperparameters

distill_lambda	distill_temp	Final Avg Acc	Sum of Forgetting
0.1	1	0.2305	3.71
0.1	2	0.2102	3.79
0.1	5	0.1949	3.88
0.1	10	0.2070	3.79
1.0	1	0.2143	2.72
1.0	2	0.2256	2.56
1.0	5	0.2266	2.82
1.0 (chosen)	10 (chosen)	0.2350	2.51
2.0	1	0.2186	1.74
2.0	2	0.1802	1.75
2.0	5	0.1986	2.16
2.0	10	0.2016	1.83
5.0	1	0.0920	0.74
5.0	2	0.1051	0.80
5.0	5	0.1201	0.85
5.0	10	0.1896	0.57
10.0	1	0.0903	0.90
10.0	2	0.1048	1.15
10.0	5	0.1388	0.27
10.0	10	0.1808	0.79

- Fine-Tuning is the simplest but forgets heavily once new tasks arrive.

Joint is not a continual learning approach but provides a reference upper bound when retraining on all data.

List of Tables

List of Figures

References

[1] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. S. Torr, "Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence," in *Proc. ECCV*, 2018, pp. 532–547, doi:10.1007/978-3-030-01264-9_32.

[2] L. Huang, Z. An, Y. Zeng, C. Yang, X. Yu, and Y. Xu, "Exemplar-Free Class Incremental Learning via Incremental Representation," *arXiv preprint arXiv:2403.16221*, 2024.

[3] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, *et al.*, "Overcoming Catastrophic Forgetting in Neural Networks," *Proc. Natl. Acad. Sci.*, vol. 114, no. 13, pp. 3521–3526, 2017, doi:10.1073/pnas.1611835114.

[4] A. Kutalev and A. Lapina, "Stabilizing Elastic Weight Consolidation Method in Practical ML Tasks and Using Weight Importances for Neural Network Pruning," *arXiv preprint arXiv:2109.10021*, 2021.

[5] Z. Li and D. Hoiem, "Learning without Forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, 2017, doi:10.1109/TPAMI.2017.2773081.

[6] V. Meshram and K. Patil, "FruitNet: Indian Fruits Dataset with Quality (Good, Bad & Mixed)," *Mendeley Data*, V1, 2021, doi:10.17632/b6fftwbr2v.1.

[7] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental Classifier and Representation Learning," in *Proc. CVPR*, 2017, pp. 2001–2010, doi:10.1109/CVPR.2017.587.

[8] E. C. G. Yildirim, M. O. Yildirim, M. Kilickaya, and J. Vanschoren, "Adaptive Regularization for Class-Incremental Learning," *arXiv preprint arXiv:2304.13327*, 2023.