An Ensemble Approach to Open World Recognition in Image Classification

Antonio Albanese
Politecnico di Torino
s282043@studenti.polito.it

Leonardo Maggio
Politecnico di Torino
s292938@studenti.polito.it

Ionut Cosmin Nedescu
Politecnico di Torino
s292495@studenti.polito.it

Abstract

While Convolutional Neural Networks have brought major advances in computer vision, modern robots are not yet ready to act in the open world. Their abilities are in fact limited to closed world scenarios, in which it is assumed that the semantic concepts a model has to recognize is limited to the number of classes seen during training. One of the main problems of artificial intelligence is precisely to break the closed world assumption and make robots capable not only of recognizing what they do not know, but also making them capable of dynamically extending their knowledge about the world.

In this work, we face the incremental learning challenges, building a model that reproduces the baselines of existing classification frameworks. In a subsequent ablation study, we experiment with the combination of different components, to understand their individual contribution and verify their effectiveness. Next, we take into account the open world recognition problem and incorporate into our model a naive rejection strategy for unknown classes. Finally, we propose our rejection strategy, based on deep ensembles, which addresses the uncertainty of the model.

1. Introduction

Implementing autonomous agents capable of supporting humans in everyday life is one of the main goals of Artificial Intelligence. For this purpose, robots must be able to understand their surroundings, and research has made great advances in developing visual systems capable of understanding the rich information provided by cameras. Today we can rely on home service robots, clinical decision support systems for medical diagnoses and even self-driving cars. All these systems have in common the use of deep learning algorithms and, in particular, the use of Convolutional Neural Networks to infer knowledge from images. The most common task is classification, which is to recognize the category (class) of an image.

Most deep learning approaches rely on the closed world assumption. Specifically, it consists in assuming that (a) training and test data come from the same underlying distribution, i.e., domain shift, and (b) the set of classes seen during training constitute the only classes that will be seen at test time, i.e., semantic shift [12]. This is reflected in the performance drop of deep visual models when encountering classes or scenarios unseen during training.

Unfortunately, no matter how much knowledge can be encoded in a robot, it will inevitably see unknown objects. In order to make the agent autonomous in the open world, it must be able to: i) learn new classes incrementally (i.e., incremental learning); ii) recognize when classes do (or do not) belong to the knowledge it already has (i.e., open set); and iii) add these classes to its knowledge once data for these categories is provided [2]. Within this context, the problem of catastrophic forgetting arises [14]. A model that learns new classes over time is bound to forget the classes previously learned.

In this work, we will initially address the challenges of incremental class learning, implementing, and studying the knowledge distillation strategy. We will build a model that reproduces the baselines of existing framework to show the effects of catastrophic forgetting and gain a deeper knowledge of the phenomenon. In the subsequent ablation study, we will analyze the contribution of the individual components to the model's performance and experiment with alternatives. Next, we will introduce a naive rejection strategy to compare the model's behavior in the recognition task in the closed world and the open world. Finally, we will present our novel rejection strategy, which exploits the uncertainty of the model through deep ensembles.

2. Method

In this section, we describe the experimental setting of our work. Section 2.1 presents the dataset used, and its partition to perform the tasks of incremental class learning and open world recognition. Section 2.2 explains the components of the model and the underlying convolutional neural network architecture adopted. Section 2.3 describes the training and hyperparameter setting adopted during experiments.

2.1. Dataset

CIFAR-100 [9] has become a standard benchmark for comparing incremental class learning algorithms. It contains 60,000 tiny, labeled images, 32x32 in size. The default partition provides 50,000 images for the training set and 10,000 for the testing set. In particular, we allocate 100 images per class (20% of the training set) to form the validation set, which is used to choose the optimal weight configuration during training. We perform on the training set the same data augmentation process that is carried out in iCaRL [15], to avoid overfitting and improve our results.

In this work, the dataset is used in two different ways, depending on the task. For the incremental class learning task, we replicate the approach of iCaRL. We randomly divide the dataset into ten groups of ten classes each, and each single group is learned during a different training step. The testing set contains instead all the classes seen up to the current step. For the open world rejection task, we follow the implementation of BDOC [3], dividing the dataset into 50 known and 50 unknown classes. Thus, the first half represents the closed world set and the second the open world set. Again, the model learns new classes in steps of ten.

To have a fair benchmark, we run the experiments three times for each method and present the results by averaging the results among each run.

2.2. Model

In this work we make a simplification and use a model composed of two components, a feature extractor, and a classifier. Commonly, the feature extractor is a Convolutional Neural Network, so we adopt a Resnet-32 [5], as described in iCaRL. On the other hand, we use different classifiers and, for each experiment, we provide a description of the one employed.

2.3. Setting

We train the model over 70 epochs using Stochastic Gradient Descent with momentum as optimization algorithm. We use the hyperparameters shown in Table 1 which are the same as iCaRL, with the exception of the learning rate which we lower from 2 to 1.

Table 1: Hyperparameter setting

Hyperparameter	Value
Batch Size	128
Learning Rate	1
Momentum	0.9
Weight Decay	1e-05
Epochs	70
Milestones	[43, 63]
Gamma	0.2

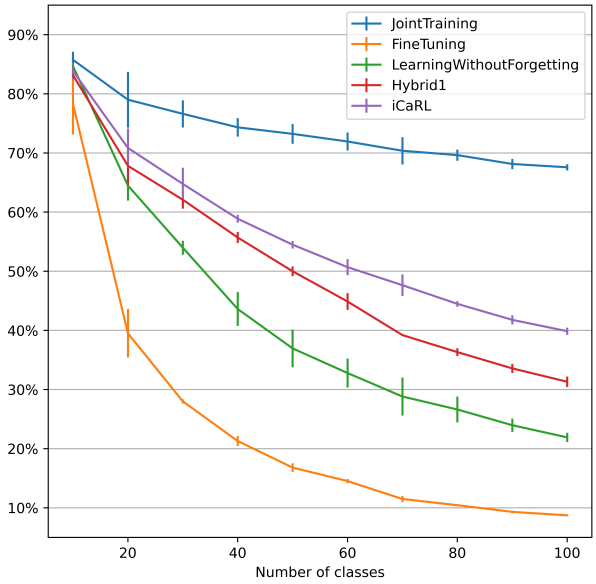


Figure 1: Test accuracy trend of different baselines and upper bound. iCaRL clearly outperforms all other incremental learning approaches.

3. Incremental Class Learning Baselines

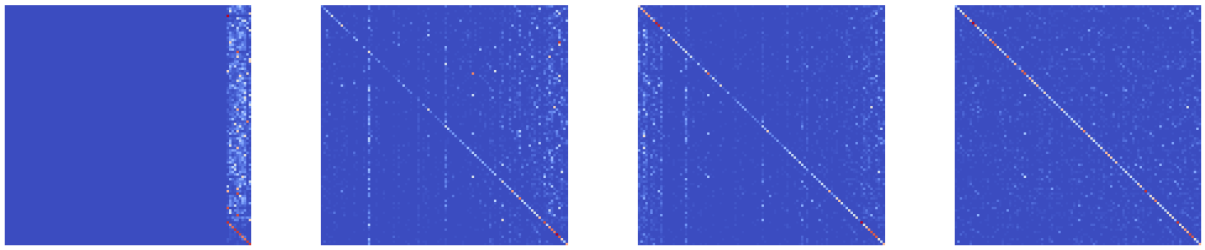
In this section, we introduce the phenomenon of catastrophic forgetting by describing our implementation of existing incremental class learning baselines, and presenting the results obtained. In particular, we repeat the experiments of the iCaRL paper, using the binary cross entropy (BCE) as classification loss. Figure 1 shows the test accuracy trend of the different models.

3.1. Joint Training

Following previous literature [7][13], we perform Joint Training as the first experiment. This approach violates the incremental learning protocol, because at each step the network is trained on the union of all the data groups seen up to that point. Catastrophic forgetting is avoided, and this baseline gives us an upper bound of reference for our next incremental learning models ($avg\_acc = 73.7\%$, see Figure 1).

3.2. Fine Tuning

The first real baseline of incremental learning that we implement is the fine tuning of the network on the new classes. There is no mechanism that preserves past knowledge, so the network adjusts its weights for the new groups of images, and becomes inadequate to recognize the old ones. This is the most severe form of catastrophic forgetting (see Figure 2a) and the least effective method among all the baselines ($avg\_acc = 23.8\%$, see Figure 1).



(a) Fine Tuning (b) LwF (c) Hybrid1 (d) iCaRL

Figure 2: Confusion matrices of different incremental class learning baselines on CIFAR-100. (a) Fine Tuning is able to predict only the classes of the last group. (b) Learning without Forgetting tends to predict classes from the last group more frequently. (c) Hybrid1's predictions are biased towards classes from the first and last group. (d) iCaRL makes predictions uniformly over all classes.

3.3. Learning without Forgetting

We use knowledge distillation to prevent that information in the network deteriorates too much over time. This technique was originally proposed in [6], and applied for the first time in an incremental learning context in [11]. The idea is to transfer the knowledge gained from the network in the previous step to the current network. At the end of each training step, the network is saved and used in the next step to make predictions on the new group of classes. The current network uses the predictions of the old network as targets for the training. Then, the distillation loss encourages the current network to reproduce the predictions of the old. In particular, we use the BCE as distillation loss. This addition restricts the effect of catastrophic forgetting (see Figure 2b) and brings improvements, compared to fine tuning ($avg\_acc = 41.8\%$, see Figure 1).

3.4. iCaRL

iCaRL uses knowledge distillation and adds two novel components, which are the use of exemplars and the nearest mean-of-exemplars (NME) classifier.

To show the improvements made by the individual components, we first reproduce the *hybrid1* experiment from iCaRL, which uses only the exemplars. The model is allowed to store up to K = 2000 images of previous classes in a set, which is updated whenever new classes are encountered. The selection of exemplars is random, as it performs better than the prioritized selection originally proposed. The dataset is augmented with exemplars before training, allowing the model to rehearse previously gained knowledge. This further mitigates the effects of catastrophic forgetting (see Figure 2c) and increases the average test accuracy ($avg\_acc = 50.4\%$, see Figure 1).

iCaRL uses the NME classifier in place of the fully-connected layer of *hybrid1* during testing. It calculates the prototype vector for each class observed so far and the feature vector of the image to classify, then assigns it the label of the most similar vector. Replacing the classifier makes the model more robust and results in an additional increase in testing accuracy (see Figure 1). Calculating the average of the features extracted at each training step allows us to correctly classify even images from previous groups, and mitigate the effects of catastrophic forgetting ($avg\_acc = 55.7\%$, see Figure 2d).

4. iCaRL Ablation Study

In this section we use the iCaRL model obtained in 3.4 and change some components to study its effects. Section 4.1 experiments with the combination of different classification and distillation losses. Section 4.2 analyzes possible alternatives to the NME classifier.

4.1. Losses

We want to find an alternative to the BCE loss used so far as classification and distillation loss. Thus, we decide to use a cross entropy as classification loss and test different distillation losses. We lower the learning rate from 1 to 0.1.

Hou et al. implement in [7] the *less forget constraint* as distillation loss, to enforce a stronger constraint on the previous knowledge. It consists in maximizing the cosine similarity between the normalized features extracted by the old model, $\bar{f}^*(x)$, and those by the current model, $\bar{f}(x)$, as follows:

$$L_{dis}^{G}(x) = 1 - \langle \bar{f}^*(x), \bar{f}(x) \rangle \quad (1)$$

In this way it encourages the orientation of the features extracted from the current model to be similar to those from the old model. Moreover, they define the following weight

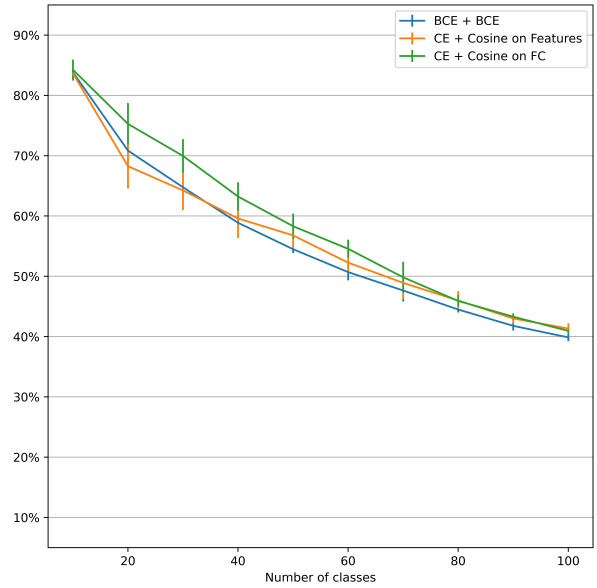


Figure 3: Test accuracy trend of the iCaRL framework with different implementation of cosine similarity loss.

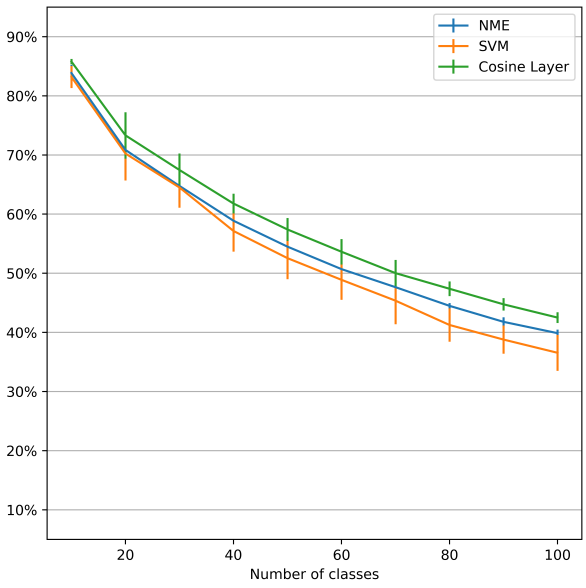


Figure 4: Test accuracy trend of the iCaRL framework with different classifiers.

λ in order to adapt the loss to the number of classes introduced in each training step and vary the knowledge to be preserved:

$$\lambda = \lambda_{base}\sqrt{\|\mathcal{C}_n\|/\|\mathcal{C}_o\|} \tag{2}$$

where $\|\mathcal{C}_n\|$ and $\|\mathcal{C}_o\|$ are the number of new and old classes in each phase, and λ_{base} is a fixed constant for each dataset. We replicate the experiment and find that $\lambda_{base} = 2$ is optimal for the current task.

Then, we test a more intuitive approach. We eliminate the weight of the loss and compute the cosine similarity between the outputs of the fully-connected layer of the old network and the current one, instead of on the extracted features. Figure 3 shows the results of this experiment and compares them with those of the previous one and of iCaRL. The *less forget constraint* performs better than the BCE as the number of classes increase ($avg\_acc = 56.4\%$), while our proposal outperforms both of them on all groups ($avg\_acc = 58.6\%$).

Given the success of this experiment, we also test in the same circumstances L1 and L2 loss in place of the cosine similarity. However, the results are not as satisfactory and for this reason we report them in the Appendix A.

4.2. Classifiers

We want to find an alternative to the NME classifier, or at least find a way to improve it. To compare the results of the following experiments with iCaRL, we leave the hyperparameters and losses unchanged.

4.2.1 Support Vector Machine

Support Vector Machines (SVM) are supervised learning models that perform classification by constructing separation hyperplanes between data points of different classes. The algorithm attempts to maximize the hyperplane margin, which is the distance between the hyperplane and the closest points of each class. The choice of the hyperplane is made considering a subset of features, so the model can work well even with a reduced number of samples. Thus, we choose to fit the SVM on the features extracted from the exemplars set at the end of each training step. In particular, we use a linear kernel and perform a grid search to find the best value of C among 0.1, 1, 10, and 100. C is a regularization parameter that controls the trade-off between smooth decision boundary and classifying the training points correctly. However, Figure 4 shows that SVM trained with exemplars does not perform as well as NME, especially when the number of classes to predict grows ($avg\_acc = 53.8\%$).

4.2.2 Cosine Linear Layer

The standard fully-connected layer computes the classification scores using an inner product between its weights vector θ_i and the vector containing the features extracted $f(x)$:

$$\theta_i^T f(x) + b_i \tag{3}$$

where b_i is the bias.

Several studies show that the fully-connected layer is biased towards the new classes, due to the different proportion

between the number of new images and the exemplars, and this is a crucial cause of catastrophic forgetting [1][7][16]. We adopt the solution proposed in [7] and implement a cosine normalization layer in place of the fully-connected. It computes the cosine similarity between its weights and the features extracted from the network to assign scores to the input images. The difference between the magnitude of the weight vectors of the old and new classes is reduced by containing both of them within a hypersphere (see Figure 5).

Compared to Equation 3, the cosine similarity normalizes both the features extracted and the weights, before computing the inner product:

$$\eta \langle \bar{\theta}_i, \bar{f}(x) \rangle \tag{4}$$

where $\bar{v} = v/\|v\|_2$ denotes the l_2-normalized vector, and $\langle \bar{v}_1, \bar{v}_2 \rangle = \bar{v}_1^T \bar{v}_2$ measures the cosine similarity between two normalized vectors. The learnable scalar η is introduced to control the peakiness of softmax distribution since the range of $\langle \bar{v}_1, \bar{v}_2 \rangle$ is restricted to $[-1, +1]$.

Figure 4 shows that the cosine layer is a successful variation that improves model training and thus the performance of the NME classifier ($avg\_acc = 58.4\%$).

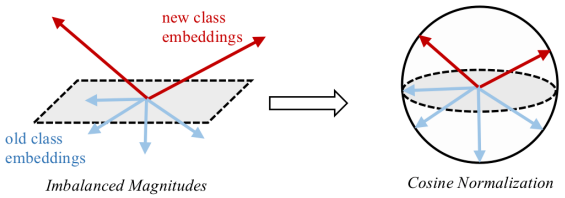


Figure 5: Cosine similarity helps to contain the class weights of old and new classes inside a hypersphere [7].

5. Open World Recognition

Having addressed the incremental class learning problem, we now move on to the open world recognition task, which takes place as described in 2.1. Section 5.1 presents the task of rejecting unknown samples and the comparison metrics for OWR methods. Section 5.2 describes our first naive approach, while Section 5.3 illustrates our novel rejection strategy, based on deep ensembles.

5.1. Metrics

We take the same approach as BDOC [3], which runs OWR methods on 3 different scenarios, then uses 3 standard metrics to compare their performance. The closed world scenario without rejection consists in testing the model on known classes and measuring the accuracy of the classification. The closed world scenario with rejection consists in testing the model on known classes and measuring the accuracy of the classification, given the additional alternative of classifying samples as *unknown*. The open world scenario consists in testing the model only on unknown classes and measuring the *harmonic mean* between the accuracy on open set and the closed world. This metric is adopted to mitigate the bias created by the accuracy computed on the open set scenario, where all samples should be rejected because they are truly unknown.

5.2. Naive Approach

We obtain the model to be used in the OWR task by combining the components that worked best in the ablation study. Therefore, we use the Cosine Layer together with the combination of CE loss and Cosine Similarity loss, resulting in a model that outperforms all the previous ($avg\_acc = 59.2\%$ on 100 classes, $avg\_acc = 71.3\%$ on 50 classes).

Our naive model applies a threshold to each probability obtained from the softmax. If it is higher, the model is considered highly confident, and the class is considered to be known. If it is lower, the model is considered as slightly confident, and the class is considered as unknown. Table 2 shows the accuracy obtained for different thresholds on the closed word with rejection and Table 3 the harmonic mean.

Table 2: Accuracy on the closed world with rejection scenario for different thresholds.

	1	2	3	4	5
t=0.50	0.86	0.76	0.68	0.60	0.53
t=0.75	0.80	0.71	0.64	0.56	0.51
t=0.95	0.72	0.63	0.56	0.47	0.42

Table 3: Harmonic mean between the accuracy on open set and the closed world for different thresholds.

	1	2	3	4	5
t=0.50	0.12	0.15	0.16	0.16	0.15
t=0.75	0.39	0.39	0.38	0.37	0.36
t=0.95	0.63	0.56	0.53	0.49	0.46

The results obtained show that our model behaves very differently on the two tasks. The threshold that allows us to perform best is 0.95. A model that works well with a lower threshold in the closed world scenario shows a drop in performance in the open world. Higher thresholds are needed in the open world scenario, and we believe that this is due to a bad calibration of the network. Studies show how deep and strongly regularized architectures tend to have a distribution of predicting probabilities highly skewed towards 0 or 1 [4]. This can make the model overconfident even when unseen test example are presented, and the absolute value of these probabilities could be meaningless.

5.3. Our Ensemble Approach

In this section we want to improve the rejection strategy of our naive model and, at the same time, address the problem of model overconfidence. For this purpose, we implement a deep ensemble like those introduced in [10], made with 5 base estimators defined as the one used in 5.2. Ensembles work according to the hypothesis that combining multiple models together can often produce a much more powerful model, able to outperform every single component.

We train the model using the snapshot technique to force the base estimators to settle in different local minima [8]. The idea is to obtain re-calibrated predictions that better reflect the true confidence of the model, averaging those of the 5 models. Together with more robust predictions, deep ensembles also allow us to calculate the uncertainty of the model, by means of the standard deviation of the predictions.

We then carry out two experiments.

(a) In the first, we use the same rejection strategy of the naive approach and compare the average of the predictions with the threshold.

(b) In the second, we also take into account the uncertainty of the model and reject all prediction that deviate from the threshold more than one and a half time the sample standard deviation.

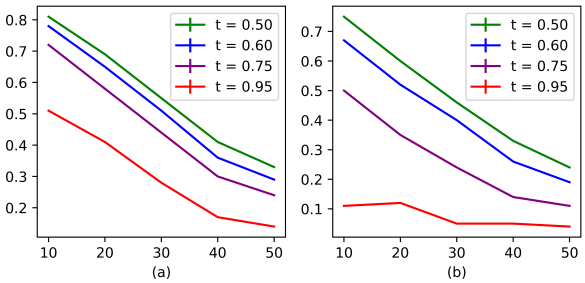


Figure 6: Accuracy on the closed world with rejection scenario for different thresholds.

Figure 6 reports the accuracy calculated with different thresholds in the closed world scenario for the two experiments. Both approaches prove to be unsuccessful, since all performances degrade much more rapidly than the naive approach. Note that the model now requires lower thresholds in order to work, confirming the fact that it is no longer so confident.

Tables 4 and 5 report the harmonic means calculated with the two rejection strategies. Again, the results are rather unsuccessful when compared with those of Table 3. The best performing model adopts the rejection strategy (a) and uses a threshold equal to t = 0.75, but does not surpass the naive approach.

Table 4: Harmonic mean between the accuracy on open set and the closed world for different thresholds using the rejection strategy (a).

	1	2	3	4	5
t=0.50	0.31	0.39	0.40	0.39	0.37
t=0.60	0.46	0.49	0.47	0.42	0.38
t=0.75	0.59	0.57	0.51	0.41	0.36
t=0.95	0.62	0.54	0.42	0.28	0.24

Table 5: Harmonic mean between the accuracy on open set and the closed world for different thresholds using the rejection strategy (b).

	1	2	3	4	5
t=0.50	0.56	0.53	0.50	0.42	0.35
t=0.60	0.63	0.56	0.50	0.38	0.31
t=0.75	0.61	0.49	0.37	0.25	0.20
t=0.95	0.20	0.21	0.09	0.10	0.14

6. Conclusion

In this paper we presented the problem of open world recognition, explaining that, to make an agent autonomous, it must perform the task of incremental class learning and must be able to recognize unknown objects. We have built existing incremental class learning baselines to explain the phenomenon of catastrophic forgetting. In the ablation study, we found the best components for our model, which we subsequently took into the open world to recognize unknown images. Finally, we have proposed our own rejection strategy that takes into account the uncertainty of the model, using deep ensembles. However, our proposal was unable to outperform the naive approach.

For more details, we invite readers to consult the following repository where the code is available: `https://github.com/cosminnedescu/ProjectMLDL`.

References

[1] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 583–592, 2019.

[2] Abhijit Bendale and Terrance Boult. Towards open world recognition. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015.

[3] Dario Fontanel, Fabio Cermelli, Massimiliano Mancini, Samuel Rota Buló, Elisa Ricci, and Barbara Caputo. Boosting deep open world recognition by clustering. *IEEE Robotics and Automation Letters*, 5(4):5985–5992, 2020.

[4] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks, 2017.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[6] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[7] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[8] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E. Hopcroft, and Kilian Q. Weinberger. Snapshot ensembles: Train 1, get m for free, 2017.

[9] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

[10] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles, 2017.

[11] Zhizhong Li and Derek Hoiem. Learning without forgetting, 2017.

[12] Massimiliano Mancini, Zeynep Akata, Elisa Ricci, and Barbara Caputo. Towards recognizing unseen categories in unseen domains, 2020.

[13] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation on image classification, 2021.

[14] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press, 1989.

[15] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. icarl: Incremental classifier and representation learning, 2017.

[16] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning, 2019.

A. iCaRL Losses Ablation Study

We repeat the experiment of Section 4.1, using L1 and L2 as distillation. L1 loss measures the mean absolute errors between the input and the target. L2 measures the mean squared error between the input and the target. We apply them both between the normalized features extracted from the networks and between the outputs of the fully-connected layers. In the first case, we also calculate the weight λ.

The results for L1 and L2 are shown in Figures 7 and 8, respectively. Applying the distillation between the outputs does not work with these two losses, and causes the model performance to drop (L1 $avg\_acc = 53.3\%$, L2 $avg\_acc = 55.1\%$). On the other hand, the less forget constraint approach brings improvements over the iCaRL baseline (L1 $avg\_acc = 58.2\%$, L2 $avg\_acc = 58.0\%$). These last results are in line with that obtained with the cosine similarity and could be due to the weight λ, which made the distillation component less incident as the classes increased and allowed the curve to flatten.

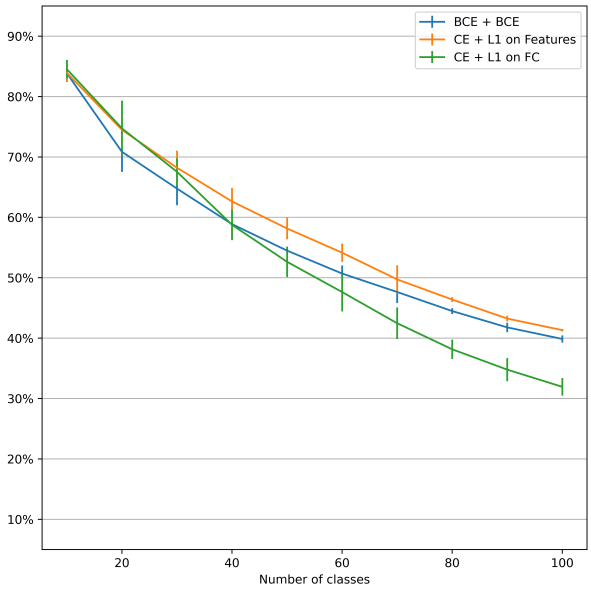


Figure 7: Test accuracy trend of the iCaRL framework with different implementation of L1 loss.

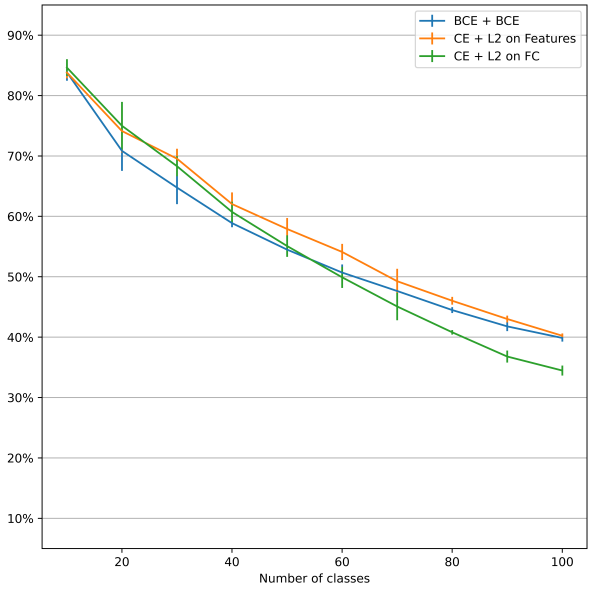


Figure 8: Test accuracy trend of the iCaRL framework with different implementation of L2 loss.