Supplementary material for "An Analysis of Initial Training Strategies for Exemplar-Free Class-Incremental Learning"

Grégoire Petit*[1,2], Michael Soumm*[1], Eva Feillet*[1,4], Adrian Popescu[1],
Bertrand Delezoide[3], David Picard[2], Céline Hudelot[4]
[1]Université Paris-Saclay, CEA, LIST, F-91120, Palaiseau, France
[2]LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France
[3]Amanda, 34 Avenue Des Champs Elysées, F-75008, Paris, France
[4]Université Paris-Saclay, CentraleSupélec, MICS, France
g.petit360@gmail.com, {michael.soumm, eva.feillet, adrian.popescu}@cea.fr,
david.picard@enpc.fr,bertrand.delezoide@amanda.com,celine.hudelot@centralesupelec.fr

In this supplementary material, we provide details regarding:

1. the datasets used in our experiments,
2. the implementation of incremental learning and pre-training algorithms,
3. the method for selecting linear regression models,
4. the analysis of different factors on the accuracy of incremental learning models.

1. Target datasets

We experiment with a wide variety of datasets in terms of domain, granularity, number of samples per class, and complexity of patterns to recognize. We select thirteen datasets containing 100 classes and three datasets containing 1000 classes as follows. The datasets $IMN100_1$ and $IMN100_2$ are obtained by randomly sampling 100 classes from ImageNet-21k [2] which are not present in ILSVRC [11]. Flora is a thematic subset of ImageNet obtained by sampling 100 classes under the concept 'flora', without intersection with ILSVRC. We also used 100-classes subsets of WikiArt [12] (Art100), Casia-align [16] (Casia100), Food101 [1] (Food100), FGVC-Aircraft [8] (Air100), MTSD [7] (MTSD100), Google Landmarks v2 [15] (Land100), Logo2K [14] (Logo100) and Quickdraw [3] (Qdraw100). We build two fine-grained subsets from iNaturalist [13] (2018 version) by selecting (i) amphibia species (Amph100) and (ii) fungi species (Fungi100) which do not intersect with the ILSVRC dataset. Finally, we also use three 1000-classes subsets of Casia-align (Casia1k), Google Landmarks v1 [9] (Land1k), and iNaturalist (iNat1k), respectively.

The average number of images per dataset is reported in Table 1. For reproducibility purposes, we will provide in a repository the distribution of images between the training and test subsets of each dataset, as well as the distribution of classes between the steps of the incremental learning process.

Dataset	μ_{train}	μ_{test}	σ_{train}	σ_{test}
Casia100	250.0	50.0	0.0	0.0
Food100	750.0	250.0	0.0	0.0
Land100	300.0	50.0	0.0	0.0
$IMN100_1$	340.0	60.0	0.0	0.0
$IMN100_2$	340.0	60.0	0.0	0.0
Flora	340.0	60.0	0.0	0.0
Logo100	80.0	15.0	0.0	0.0
Qdraw100	500.0	100.0	0.0	0.0
Art100	150.0	25.0	0.0	0.0
MTSD100	100.0	20.0	0.0	0.0
Air100	80.0	20.0	0.0	0.0
Fungi100	300.0	10.0	0.0	0.0
Amph100	300.0	10.0	0.0	0.0
Land1k	374.37	20.0	103.83	0.0
iNat1k	300.0	10.0	0.0	0.0
Casia1k	60.0	28.0	0.0	0.0

Table 1. Number of images per class in the train and test subsets of each target dataset. The average and the standard deviation of the number of images per class are denoted by μ and σ respectively.

2. Implementation

2.1. Incremental learning algorithms

We will release the code for reproducing our experiments.

BSIL. Our implementation of LUCIR [5] algorithm with a Balanced Cross-Entropy loss [6] is based on the original repository of [5][1]. LUCIR was initially proposed as a CIL algorithm with rehearsal. In practice, as we focus on EFCIL, we set the size of LUCIR's memory buffer to zero.

DSLDA. Our implementation is based on the original repository of [4][2].

FeTrIL. Our implementation is based on the original repository of [10][3].

2.2. Pre-training algorithms

The pre-trained models are taken from the repositories indicated in the footnotes: DINOv2[4], BYOL[5], and DeiT[6]. We also used the method MoCov3[7] for training models with a ResNet50 architecture in a self-supervised manner on the initial data subset of each target dataset.

2.3. Fine-tuning

We use PyTorch[8] implementation of ResNet50 architecture and the ViT-Small transformer architecture from the checkpoints of DINOv2[4] and DeiT[6] we introduced in Subsection 2.2. When fine-tuning the models, in the case of ResNet50, we freeze the first 3 convolutional blocks and only update the parameters belonging to the last convolutional block, as well as the linear layer. In the case of ViT-Small, we freeze the blocks up to block 8 and update the blocks 9 to 11, as well as the linear layer. In both cases, the parameters are updated using a learning rate equal to one-tenth of the value of the base learning rate used to pre-train the model.

3. Linear Regression

3.1. Variable selection

We use the Python module `statsmodels` for our linear regressions. We first consider a broad range of explanatory variables:

- Acc_1: the accuracy of the first state,
- $Data$: dummy variable for the type of target dataset,
- $Train$: dummy variable for the initial training strategy,
- $Incr$: dummy variable for the incremental method used,

Variable	*p-value*	R^2
Acc_1	2.96e-240	0.63
$Train$	1.17e-87	0.33
$Data$	2.25-55	0.23
$Incr$	7.52e-29	0.11
n_{mean}	8.16e-20	0.07
$Small$	1.84e-05	0.02
$Width$	9.78e-03	0.01
B	1.05e-01	0.00
N	2.41e-01	0.00
N_1	2.87e-01	0.00

Table 2. Variables predicting accuracy, sorted by decreasing importance

- n_{mean}: the mean number of images per class in the experiment,
- $Small$: binary variable encoding if the training images are so small that they have to be up-scaled,
- $Width$: mean width of the images used for the experiment,
- B: binary variable encoding for the 2 possible CIL scenarios (i.e. either 10% or 50% of the total number of classes learned in the initial step of the process),
- N: the total number of classes,
- N_1: the number of images in the first state.

It has to be noted that some of these variables are highly collinear with each other since they are properties of the dataset of the experiment.

We first perform 1-variable regressions of the incremental accuracy $\overline{Acc}$ and the forgetting F. We identify the most important variables by looking at the R^2 of the regressions that have a sufficiently small $p-value$ (at the .05 threshold). Results are presented in tables 2 and 3. We select the four most important variables and use them to fit more complex linear regression models that combine these selected variables.

3.2. Model selection

We perform linear regressions with many different combinations of the selected variables. We find that introducing product variables, such as $Train \times Incr$ with the intent of directly modeling the interactions between the initial training strategy and the incremental method, introduces collinearity problems. Therefore, we choose to study such interactions following the protocol presented in Section 5 of the main paper.

[1] https://github.com/hshustc/CVPR19_Incremental_Learning
[2] https://github.com/tyler-hayes/Deep_SLDA
[3] https://github.com/GregoirePetit/FeTrIL
[4] https://github.com/facebookresearch/dinov2
[5] https://github.com/yaox12/BYOL-PyTorch
[6] https://github.com/facebookresearch/deit
[7] https://github.com/facebookresearch/moco-v3/tree/main
[8] https://pytorch.org/vision/main/_modules/torchvision/models/resnet.html#resnet50

Variable	*p-value*	R^2
$Incr$	2.20e-222	0.62
$Train$	6.46e-15	0.08
Acc_1	7.71e-10	0.03
$Data$	2.66e-03	0.02
N	7.50e-04	0.01
B	3.43e-02	0.00
N_1	4.13e-02	0.00
n_{mean}	1.07e-01	0.00
$Small$	6.88e-01	0.00
$Width$	7.17e-01	0.00

Table 3. Variables predicting forgetting, sorted by decreasing importance

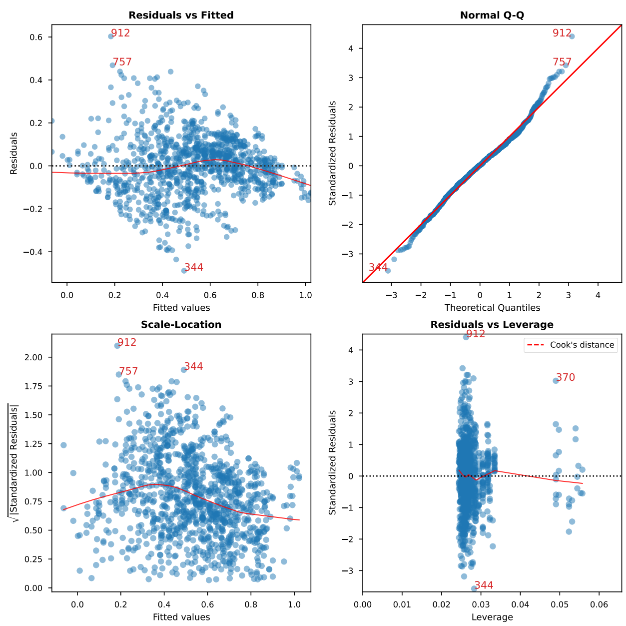

Figure 1. Diagnostics of the regression for the accuracy as in Equation 1.

We select the following model:

$$\overline{Acc} \sim Incr + Train + Data. \quad (1)$$

The output of the regression is shown in Figure 7. To verify the quality of the regression, we also plot the residuals along with a Q-Q plot to verify their normality, as well as a scale-location plot to verify homoscedasticity (constant variance), and a residual vs. leverage plot to look for possible influential outliers. All of these diagnostics are shown in Figure 1.

4. Influence of factors on accuracy

Let us recall the overall pairwise comparisons in Figure 3. We explore the effects of other variables by splitting the data with respect to a studied variable and report the regression results separately.

- Figure 4 presents the results for each target dataset,
- Figure 5 presents the results for each incremental algorithm,
- Figure 6 presents the results depending on the number of classes in the initial state.

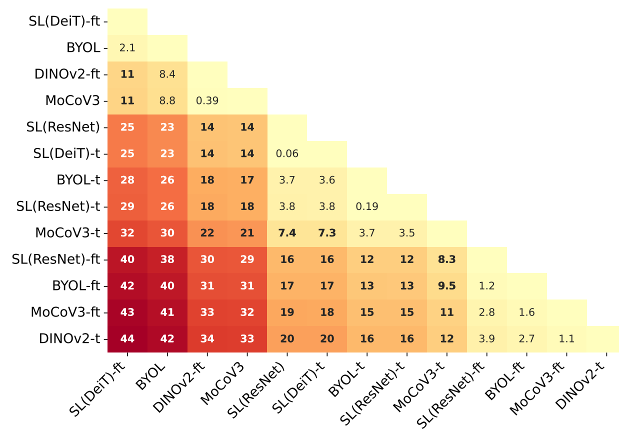

Figure 2. Overall pairwise comparisons on $\overline{Acc}$

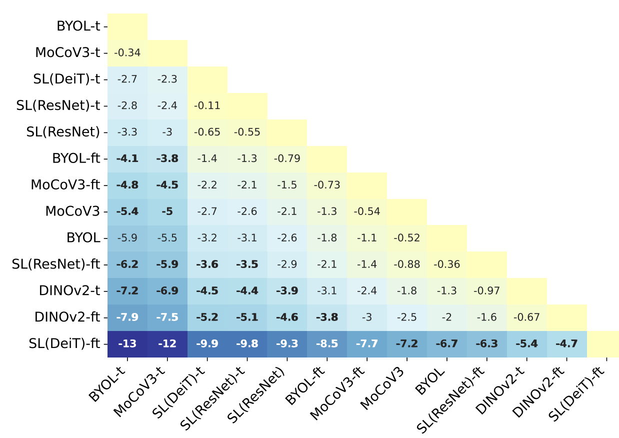

Figure 3. Overall pairwise comparisons on Forgetting

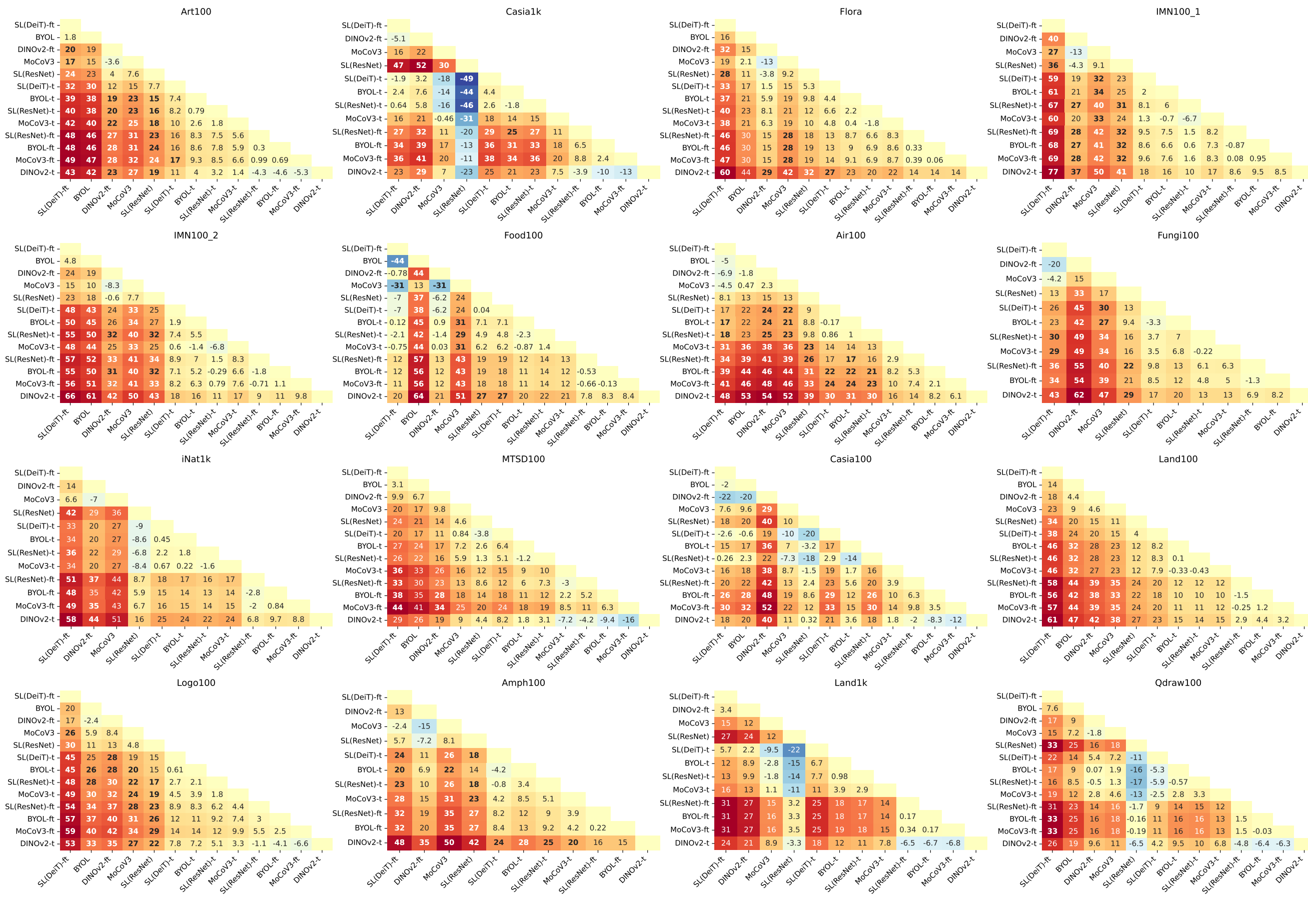

Figure 4. Pairwise gain of accuracy per dataset

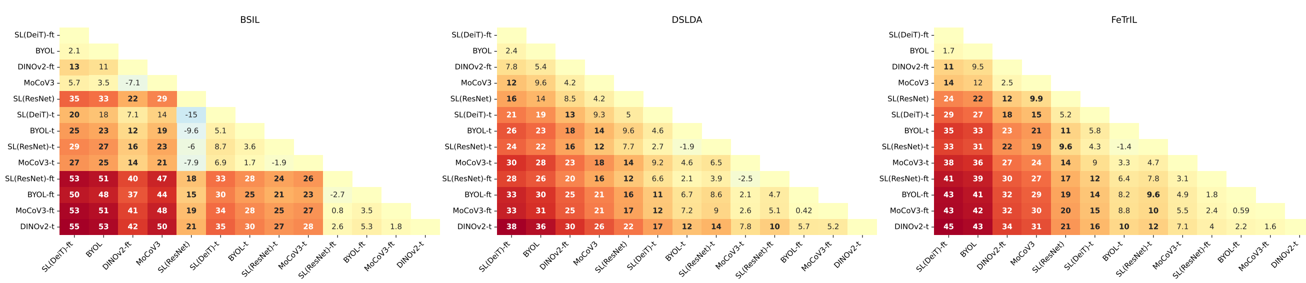

Figure 5. Pairwise gain of accuracy per method

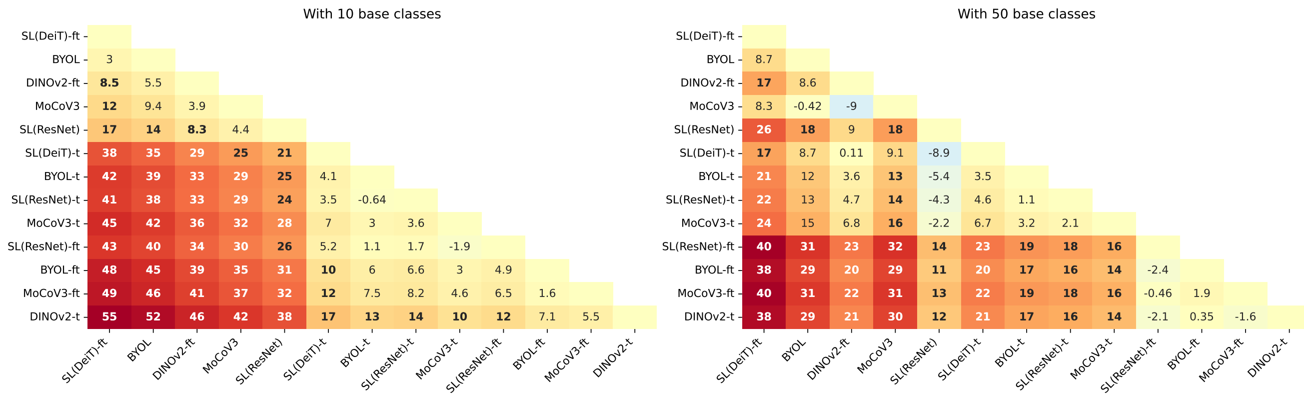

Figure 6. Pairwise gain of accuracy per number of classes in the initial state

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      A   R-squared:                       0.688
Model:                            OLS   Adj. R-squared:                  0.679
Method:                 Least Squares   F-statistic:                     80.78
Date:                Fri, 30 Jun 2023   Prob (F-statistic):          2.94e-245
Time:                        08:13:21   Log-Likelihood:                 624.55
No. Observations:                1094   AIC:                            -1189.
Df Residuals:                    1064   BIC:                            -1039.
Df Model:                          29
Covariance Type:            nonrobust
==============================================================================================================================
                                                                             coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------------------------------------
Intercept                                                                  0.0861      0.031      2.804      0.005       0.026       0.146
C(D)[T.casia1000]                                                         -0.1287      0.023     -5.569      0.000      -0.174      -0.083
C(D)[T.fgvc-aircraft-2013b]                                               -0.0841      0.023     -3.659      0.000      -0.129      -0.039
C(D)[T.food100]                                                            0.1257      0.022      5.601      0.000       0.082       0.170
C(D)[T.imagenet_flora]                                                     0.0687      0.023      3.032      0.002       0.024       0.113
C(D)[T.imagenet_random_3]                                                  0.1986      0.023      8.728      0.000       0.154       0.243
C(D)[T.imagenet_random_4]                                                  0.1821      0.022      8.114      0.000       0.138       0.226
C(D)[T.inat1000]                                                           0.0570      0.024      2.425      0.015       0.011       0.103
C(D)[T.inat_amphibia100]                                                  -0.1576      0.025     -6.224      0.000      -0.207      -0.108
C(D)[T.inat_fungi100]                                                      0.1359      0.025      5.368      0.000       0.086       0.186
C(D)[T.landmarks100]                                                       0.1580      0.022      7.118      0.000       0.114       0.202
C(D)[T.landmarks1000]                                                      0.3200      0.024     13.612      0.000       0.274       0.366
C(D)[T.logo100]                                                            0.0327      0.023      1.430      0.153      -0.012       0.078
C(D)[T.mtsd_subset]                                                        0.0660      0.023      2.873      0.004       0.021       0.111
C(D)[T.quickdraw100]                                                       0.1568      0.023      6.854      0.000       0.112       0.202
C(D)[T.wikiart100]                                                        -0.0544      0.023     -2.368      0.018      -0.100      -0.009
C(M)[T.Deep-SLDA]                                                          0.1938      0.011     18.360      0.000       0.173       0.214
C(M)[T.FeTrIL]                                                             0.1631      0.011     15.406      0.000       0.142       0.184
C(P)[T.BYOL on imagenet]                                                   0.2630      0.031      8.468      0.000       0.202       0.324
C(P)[T.BYOL on imagenet Finetuning 25% on base classes]                    0.3956      0.031     12.740      0.000       0.335       0.457
C(P)[T.DINOv2 on imagenet]                                                 0.4223      0.032     13.248      0.000       0.360       0.485
C(P)[T.DINOv2 on imagenet Finetuning 25% on base classes]                  0.0844      0.031      2.698      0.007       0.023       0.146
C(P)[T.DeiT on imagenet]                                                   0.2270      0.032      7.123      0.000       0.164       0.290
C(P)[T.DeiT on imagenet Finetuning 25% on base classes]                   -0.0210      0.031     -0.674      0.500      -0.082       0.040
C(P)[T.MocoV3 on base classes]                                             0.0884      0.031      2.817      0.005       0.027       0.150
C(P)[T.MocoV3 on imagenet]                                                 0.3003      0.031      9.613      0.000       0.239       0.362
C(P)[T.MocoV3 on imagenet Finetuning 25% on base classes]                  0.4115      0.031     13.078      0.000       0.350       0.473
C(P)[T.Supervised Learning on base classes]                                0.2264      0.031      7.290      0.000       0.165       0.287
C(P)[T.Supervised Learning on imagenet]                                    0.2648      0.031      8.496      0.000       0.204       0.326
C(P)[T.Supervised Learning on imagenet Finetuning 25% on base classes]     0.3831      0.031     12.337      0.000       0.322       0.444
==============================================================================
Omnibus:                       23.579   Durbin-Watson:                   1.401
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               45.472
Skew:                           0.080   Prob(JB):                     1.34e-10
Kurtosis:                       3.986   Cond. No.                         28.4
==============================================================================
```

Figure 7. Output of the regression for the accuracy

References

[1] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014. 1

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255, 2009. 1

[3] David Ha and Douglas Eck. A neural representation of sketch drawings. *CoRR*, abs/1704.03477, 2017. 1

[4] Tyler L Hayes and Christopher Kanan. Lifelong machine learning with deep streaming linear discriminant analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 220–221, 2020. 2

[5] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 831–839, 2019. 1, 2

[6] Quentin Jodelet, Xin Liu, and Tsuyoshi Murata. Balanced softmax cross-entropy for incremental learning. In *Artificial Neural Networks and Machine Learning–ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part II*, pages 385–396. Springer, 2021. 1

[7] Ahmed Madani and Rubiyah Yusof. Malaysian traffic sign dataset for traffic sign detection and recognition systems. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 8(11):137–143, 2016. 1

[8] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 1

[9] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *Proceedings of the IEEE international conference on computer vision*, pages 3456–3465, 2017. 1

[10] Grégoire Petit, Adrian Popescu, Hugo Schindler, David Picard, and Bertrand Delezoide. Fetril: Feature translation for exemplar-free class-incremental learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3911–3920, January 2023. 2

[11] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 1

[12] Babak Saleh and Ahmed Elgammal. Large-scale classification of fine-art paintings: Learning the right metric on the right feature. *arXiv preprint arXiv:1505.00855*, 2015. 1

[13] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018. 1

[14] Jing Wang, Weiqing Min, Sujuan Hou, Shengnan Ma, Yuanjie Zheng, Haishuai Wang, and Shuqiang Jiang. Logo-2k+: A large-scale logo dataset for scalable logo classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34 (4), pages 6194–6201, 2020. 1

[15] Tobias Weyand, Andre Araujo, Bingyi Cao, and Jack Sim. Google landmarks dataset v2-a large-scale benchmark for instance-level recognition and retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2575–2584, 2020. 1

[16] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014. 1