Few-Shot Unsupervised Continual Learning through Meta-Examples (Supplementary Material)

Anonymous Author(s)
Affiliation
Address
email

1 Test on SlimageNet64

We also make some preliminary attempts on SlimageNet64 [1], a novel and difficult benchmark for few-shot continual learning. We make the embeddings using DeepCluster [2] and we report the obtained results in Table 1. We find that our update method based on meta-example overcomes the baselines on both supervised and unsupervised approaches. FUSION-ME with 1600 clusters reaches better performances then the 800 clusters cases, meaning that a more refined partition of the embedding space is more beneficial then a rigid partitioning into classes.

Table 1: Meta-test test results on SlimageNet64 dataset.

Algorithm/Classes	5	10	20	30	40	50
Oracle OML	24.0 ±2.5.	14.3 ±3.1	15.9 ±8.3	5.0 ±1.1	**9.4 ±1.6**	2.0 ±0.4
Oracle OML-ME	**31.2 ±1.6**	**23.8 ±1.7**	**25.3 ±2.9**	**5.5 ±1.4**	7.2 ±1.3	**5.7 ±0.3**
FUSION	22.4 ±1.9	16.1 ±1.9	23.3 ±5.0	3.8 ±0.6	8.9 ±0.0	2.1 ±1.2
FUSION-ME-800	22.4 ±3.2	13.9 ±0.0	25.8 ±0.0	**4.7 ±0.3**	**9.0 ±0.1**	2.2 ±0.4
FUSION-ME-1600	**27.2 ±1.6**	**16.1 ±1.1**	**26.4 ±0.6**	4.2 ±0.6	8.1 ±1.0	**2.3 ±0.7**

2 Out-of-Distribution Tasks

Since our model is unsupervised, FEN training is only based on feature embeddings, with no class-dependent bias. This way, our model could be general enough for OoD tasks, where the training tasks belong to a different data distribution (i.e. a different dataset) with respect to the test tasks. To investigate this conjecture, we test our model on the Cifar100 and Cub datasets. Results in Table 2 show that, by training on Mini-ImageNet and testing on Cifar100 (top half) or training on Omniglot and testing on Cub (bottom half), the unsupervised approach generally outperforms the supervised one. In the latter case, FUSION-ME also outperforms the supervised oracle trained on Cub, which is incapable of learning a meaningful representation in our particular setting.

3 Rehearsal at Meta-Train Time

Rehearsal strategy can be useful at meta-test time. In particular, when the CLN is adapted to new tasks in an incremental fashion, its weights can be overridden favoring the last tasks at the expense of the first ones. The beneficial effect of rehearsal at meta-test time can be noticed when the number of test tasks is high. In fact, reservoir sampling is generally helpful on Omniglot, that is tested on 200 classes,

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

Table 2: Meta-test test results with Out-of-Distribution tasks on Cifar100 and Cub datasets.

Cifar100/Classes	2	4	6	8	10
Oracle OML Cifar100	66.0	45.0	34.0	30.0	29.5
OML-ME Mini-ImageNet	58.0	33.0	**35.3**	25.7	**24.9**
FUSION-ME Mini-ImageNet	**66.0**	**35.0**	28.7	**34.3**	22.2
Cub/Classes	2	10	20	30	40
Oracle OML Cub	50.0	13.9	25.8	4.5	8.9
OML-ME Omniglot	44.0	49.1	32.7	**27.0**	25.1
FUSION-ME Omniglot	**66.0**	**53.3**	**28.3**	26.2	**25.6**

Table 3: Meta-test test results on Omniglot dataset with rehearsal only during meta-test and both at meta-train and meta-test.

Algorithm/Classes	10	50	75	100	150	200
FUSION RS only test	67.9	55.1	46.2	37.0	29.6	**25.6**
FUSION RS both train/test	**75.9**	**56.8**	**51.2**	**39.7**	**30.5**	25.0
FUSION-ME RS only test	**81.6**	**56.4**	**54.0**	**44.6**	**34.1**	**27.4**
FUSION-ME RS both train/test	74.7	47.0	48.4	38.3	28.9	24.2

while it does not give the same benefit on Mini-ImageNet, where it reaches similar or a little lower performance. We want to verify if rehearsal can be beneficial also at meta-train time, replacing the query set $\mathcal{S}_{query}$ with a coreset built with reservoir sampling $\mathcal{S}_{coreset}$. This way, instead of sampling from random clusters, a buffer of previously seen data is stored in a buffer of fixed dimension. We try three different memory size 200, 500, 1000, obtaining, as expected, increasing results as the size increases. In Table 3 we report accuracy results on Omniglot adding rehearsal only at meta test time, and adding it at both meta-train and meta-test time with FUSION and FUSION-ME. We report only the results obtained with a buffer size 500 to avoid redundancies. As it can be noted, with FUSION, using a coreset instead of a query set at meta-train time increase the performance with respect to the case of query set usage, meaning that the representation suffers from catastrophic forgetting and the use of random data (acting only for generality purposes and not contrasting forgetting) are not enough to learn a good representation. On the contrary, with FUSION-ME, the use of a rehearsal strategy at meta-train time get worse performance. We hypothesize that this behavior is due to the different number of inner loop update between the two models. In fact, FUSION, making several inner loop updates on data belonging to the same cluster, brings the CLN weights nearest the current cluster, suffering the effect of forgetting more then FUSION-ME that makes a single inner loop on the meta-example. These results prove that, at meta-train time, FUSION-ME needs only the *generalization* ability given by $\mathcal{S}_{query}$, while FUSION needs also the *remembering* ability given by $\mathcal{S}_{coreset}$.

4 Details on Balancing Techniques

To verify the effect of unbalanced tasks during meta-training, we apply two techniques to balance tasks, one at data-level, *data augmentation* and the other at model-level, *loss balancing*. We briefly explain how these methods are implemented.

4.1 Data Augmentation

We apply data augmentation on the Omniglot dataset to observe if balancing the clusters could lead to superior performance. We notice that the results reached applying data augmentation are comparable with the one obtained with unbalanced tasks. Practically, we sample 20 elements from the clusters bigger than 20, while we exploit augmentation on the cluster with less than 20 elements. Till reaching 20 samples for tasks, we pick each time a random image between the ones in the cluster employing a

random combination of various augmentation techniques, such as horizontal flip, vertical flip, affine transformations, random crop, and color jitter. In detail, about the random crop, we select a random portion included between 75%, 80%, 85%, or 90% of the entire image. Regarding the color jitter, we use brightness, contrast, saturation, and hue factor (the first three denote a factor including between 0.8 and 1.2, the hue instead one including between -0.02 and 0.02) to adjust the image.

4.2 Loss Balancing

Our model applies clustering on all training data before starting to learn the meta-representation. This way, we can find the maximum C_{max} and minimum number C_{min} of elements per cluster obtained by k-means algorithm. Then, for each cluster, we find its number of elements $C_{current}$ and compute the balanced vector Γ as follow.

$$\Gamma = \frac{C_{max} - C_{min}}{C_{current} - C_{min} + \epsilon}, \tag{1}$$

where ϵ is used to avoid division by zero. Finally Γ is normalized as follow.

$$\Gamma_{norm} = \frac{\Gamma - \Gamma_{min}}{\Gamma_{max} - \Gamma_{min}}. \tag{2}$$

For each sampled task (taskId), the corresponding balancing parameter is selected and multiplied by the cross-entropy loss CE during meta-optimization as reported in below.

$$L = \Gamma_{norm}[\text{taskId}] \cdot CE(\text{logits}, Y), \tag{3}$$

where logits indicate the output of the model.

5 Comparison with SeLa Embeddings

We try a recent embedding learning method based on self-labeling, SeLa [3], that forces a balanced separation between clusters. In Table 4, we report the results obtained training our model with SeLa embeddings on Mini-ImageNet. The main idea, taking up what was done in DeepCluster [2], is to join clustering and representation learning, combining cross-entropy minimization with a clustering algorithm like K-means. This approach could lead to degenerate solutions such as all data points mapped to the same cluster. The authors of SeLa tried to solve this issue by adding the constraint that the labels must induce equipartition of the data, which they observe maximizes the information between data indices and labels. This new criterion extends standard cross-entropy minimization to an optimal transport problem, which is harder to optimize, exploiting traditional algorithms that scale badly when facing larger datasets. To solve this problem a fast version of the Sinkhorn-Knoop algorithm is applied.

In detail, given a dataset of N data points $I_1, \dots, I_N$ with corresponding labels $\mathbf{y}_1, \dots, \mathbf{y}_N \in \{1, \dots, K\}$, drawn from a space of K possible labels, and a deep neural network $\mathbf{x} = \Phi(I)$ mapping I to feature vectors $\mathbf{x} \in \mathbb{R}^D$; the learning objective is defined as:

$$\begin{gathered} \min_{p,q} E(p, q) \\ \text{subject to } \forall \mathbf{y} : q(\mathbf{y}|\mathbf{x}_i) \in \{0, 1\} \text{ and} \\ \sum_{i=1}^{N} q(\mathbf{y}|\mathbf{x}_i) = \frac{N}{K}. \end{gathered} \tag{4}$$

$E(p, q)$ is defined as the average cross-entropy loss, while the constraints mean that the N data points are split uniformly among the K classes and that each $\mathbf{x}_i$ is assigned to exactly one label. The objective in Equation (4) is solved as an instance of the optimal transport problem, for further details refer to the paper. DeepCluster adopts particular implementation choices to avoid degenerate solutions, but contrary to SeLa it does not force the clusters to contain the same number of samples. We empirically observe that in our setting an unconstrained approach leads to better results.

6 Time and Computational Analysis

In Table 5, we compare training time and computational resources usage between FUSION and FUSION-ME on Omniglot and Mini-ImageNet. Both datasets confirm that FUSION-ME, adopting a

Table 4: Meta-test test results on Mini-ImageNet dataset with Sela embedding.

Algorithm/Classes	2	4	6	8	10
FUSION	50.0	25.0	**18.0**	31.3	15.0
FUSION-ME 64	50.0	**35.0**	17.3	31.3	17.0
FUSION-ME 256	**64.0**	31.0	17.3	**32.5**	**18.3**

Table 5: Training time and GPU usage of FUSION-ME *vs.* FUSION on Omniglot and Mini-ImageNet.

Model	Dataset	Training time	GPU usage
FUSION	Omniglot	1h 32m	2.239 Gb
FUSION-ME	Omniglot	47m	0.743 Gb
FUSION	Mini-ImageNet	7h 44m	3.111 Gb
FUSION-ME	Mini-ImageNet	3h 58m	1.147 Gb

Table 6: Features comparison between our FUSION-ME and several works recently proposed in the literature involving continual learning and few-shot learning into the wild.

Few-shot	Unsupervised	Continual	Imbalance	OoD	Algorithm
✗	✗	✓	✗	✗	iCARL [4]
✗	✓	✓	✗	✗	CURL [5]
✓	✓	✗	✗	✗	CACTUs [6]
✓	✓	✗	✗	✗	UMTRA [7]
✓	✓	✗	✗	✗	UFLST [8]
✓	✗	✗	✓	✓	L2B [9]
✗	✓	✓	✗	✓	GD [10]
✓	✗	✓	✗	✗	OML [11]
✓	✗	✓	✗	✗	ANML [12]
✓	✗	✓	✗	✓	Continual-MAML [13]
✓	✗	✓	✗	✗	iTAML [14]
✓	✓	✓	✓	✓	FUSION-ME (Ours)

single inner update, trains considerably faster and uses approximately one-third of the GPU resources with respect to FUSION. This latter performs an update for each sample included in $\mathcal{S}_{cluster}$, keeping a computational graph of the model in memory for each update. This leads to slower training time, especially when the required number of epochs is high, such as for Mini-ImageNet. Even though with this kind of datasets we do not require particular GPU resources, this test shows the strength of our model in an eventually future scenario exploiting large image, deeper network, and more cluster samples.

7 Learning in the Jungle

To the best of our knowledge, a few-shot unsupervised continual learning setting has never been studied before in the literature. However, some works propose “learning in the jungle" problems, that involve a mixture of non-trivial settings. In Table 6 we compare some novel methods to our FUSION-ME, highlighting the features of each one. Our model is the only one that presents such complex setting involving few-shot learning, continual learning, unlabelled and unbalanced tasks and proposes experiments that show the model ability to learn from OoD data. Note that this analysis is not intended to be a complete analysis of all the methods of continual learning and few-shot learning, but only of those methods that have been placed in a different setting from the one that is commonly used in these two fields or that are related to them.

Table 7: Meta-test test results on Omniglot dataset with FiLM layers applied on Oracle OML.

Algorithm/Classes	10	50	75	100	150	200
Oracle OML	88.4	74.0	69.8	57.4	51.6	47.9
OML FiLM	**91.1**	**79.5**	**80.6**	**68.6**	**64.0**	**52.7**

Table 8: Meta-test test results on Cifar100 dataset with FiLM layers applied on Oracle OML trained on Omniglot.

Cifar100/Classes	2	4	6	8	10
OML Omniglot	**50.0**	**25.0**	15.3	22.8	9.4
OML FiLM Omniglot	**50.0**	**25.0**	**16.7**	**31.3**	**13.9**

8 FiLM Layers for OoD Tasks

To further improve the results testing on OoD tasks, we introduce FiLM [15] layers within the OML architecture (the supervised baseline). In a *FiLMed* neural network some conditional input is used to conditioned FiLM layers, to influence the final prediction by this input. The FiLM generator map this information into FiLM parameters, applying feature-wise affine transformation (in particular scaling and shifting) element-wise (features map-wise for CNN). If $\mathbf{x}$ is the input of a FiLM layer, $\mathbf{z}$ a conditional input, and γ and β are $\mathbf{z}$-dependent scaling and shifting vectors, the FiLM transformation is reported below.

$$\text{FiLM} = \gamma(\mathbf{z})\mathbf{x} \odot \beta(\mathbf{z}) \tag{5}$$

We apply this concept to OML, conditioning the prediction to task-specific features. We add two FiLM layers as linear layers after each of the last two convolutional layers of the FEN. These layers have adaptable parameters, updating in both the inner and the outer loop. In detail, recovering what was already done in [16], we introduce a 100-dimensional context parameter vector producing, through the linear layer, 512 filters. These filters are used to apply an affine transformation on the output of the convolutional layer. Context parameters are reset to zero before each new task, while FiLMs are trained to be general for all tasks and never reset during meta-train.

At meta-test time, we update the FiLM layers (during the meta-test train phase) and we reset the context parameters after each new task. This way, the context parameters are specific and dependent on each task while the FiLM layers can adapt themselves to the new unseen classes, in order to shift the frozen representation according to the context. This way, if a task changes, the model could be able to shift the representation reaching better generalization capabilities. The advantage is more pronounced facing with OoD tasks since their distribution is much different with respect to the meta-train one. We report some preliminary results obtained applying FiLM layers on the OML [11] model, trained on Omniglot and tested on both Omniglot (see Table 7) and Cifar100 (see Table 8). We find that OML with FiLM layers outperforms or at least equals on both dataset.

The results are promising, but we believe that much better performance could be achieved training context parameters and FiLM layers separately or introducing some tricks to train them together.

9 The Effect of Self-Attention

Here we want to empirically view how our self-attention mechanism acts on cluster images. We report some examples of clusters and the respectively self-attention coefficients that FUSION-ME associates to each image. In Figure 1 some samples obtained during FUSION-ME training are reported, on Mini-ImageNet and Omniglot respectively. The darker colors indicate the values of the highest attention coefficient, while the lighter colors indicate the lower ones. In the majority of cases, our mechanism rewards the most representative examples of the cluster, meaning the ones that globally contain most of the features present in the other examples as well. A further improvement could be to identify the outliers (the samples more distant from the others at features-level) of a

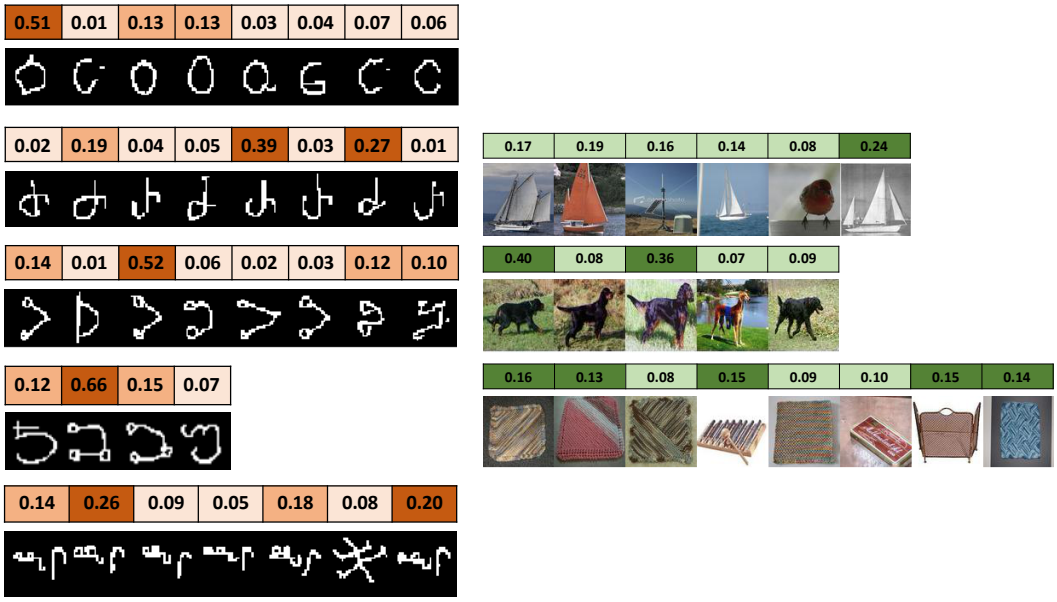

Figure 1: Samples of clusters (one for each row) generated on Omniglot (left) and Mini-ImageNet (right). Self-attention coefficients are reported associated to each image.

cluster and discard them before the self-attention mechanism is applied. This way, only the features of the correctly grouped samples can be employed to build the meta-example.

10 Datasets

To evaluate our model, we employ two standard datasets typically used to validate few-shot learning methods: Omniglot and Mini-ImageNet. In addition, we try our model on a new and challenging few-shot continual learning benchmark, SlimageNet64. The Omniglot dataset contains 1623 characters from 50 different alphabets with 20 greyscale image samples per class. We use the same splits as [6], using 1100 characters for meta-training, 100 for meta-validation, and 423 for meta-testing. The Mini-ImageNet dataset consists of 100 classes of realistic RGB images with 600 examples per class. As done in [17, 6], we use 64 classes for meta-training, 16 for meta-validation and 20 for meta-test. The SlimageNet64 dataset contains 1000 classes with 200 RGB images per class taken from the down-scaled version of ILSVRC-2012, ImageNet64x64. 800 classes are used for meta-train and the remaining 200 for meta-test purposes. Finally, we use the Cifar100 [18] and Cub [19] datasets to prove our model performance on Out-of-Distribution tasks.

11 Implementation Details

The FEN is composed of 6 convolutional layers followed by ReLU activations, 3×3 kernel (for Omniglot, the last one is a 1×1 kernel) followed by 2 linear layers interleaved by a *ReLU* activation. The attention mechanism is implemented with two additional linear layers interleaved by a *Tanh* function and followed by a *Softmax* and a sum to compute attention coefficients and aggregate features. For Omniglot, we train the model for 40000 steps while for Mini-ImageNet and SlimageNet64 for 200000, with meta-batch size equals to 1. The outer loop learning rate is set to $1e^{-4}$ while the inner loop learning rate is set to 0.01, with Adam optimizer. We report the algorithm of FUSION-ME meta-training in Algorithm 1 and an illustration of the four phases in Figure 2.

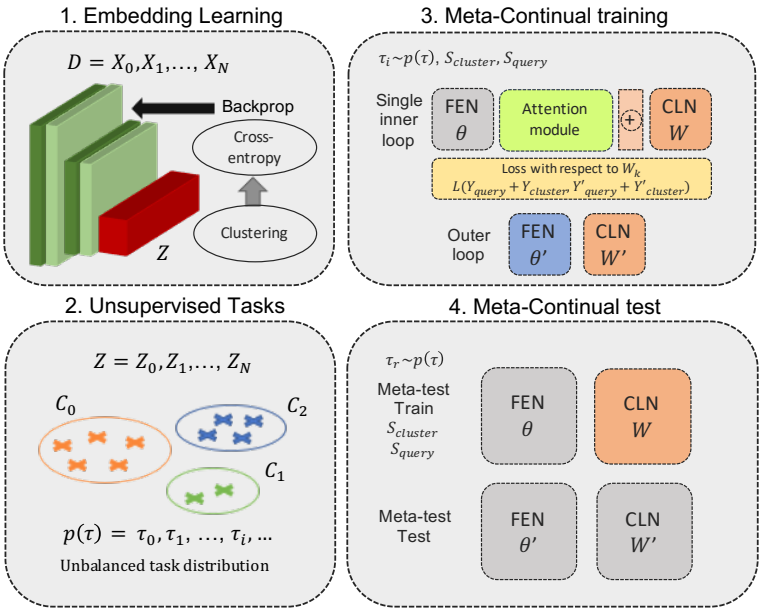

Figure 2: Scheme of FUSION-ME. The model is composed of 4 phases: embedding learning network phase, unsupervised task construction phase, meta-continual training phase and meta-continual test phase.

Algorithm 1 FUSION-ME

Require: : $D = X_0, X_1, ..., X_N$: unlabeled training set
Require: α, β: inner loop and outer loop learning rates
Run embedding learning on D producing $Z_{0:N}$ from $X_{0:N}$
Run k-means on $Z_{0:N}$ generating a distribution of unbalanced tasks $p(\mathcal{T})$ from clusters
Randomly initialize θ and W
while not done **do**
  Sample a task $\mathcal{T}_i \sim p(\mathcal{T}) = (\mathcal{S}_{cluster}, \mathcal{S}_{query})$
  Randomly initialize W_i
  $\mathcal{S}_{cluster} = \{(X_k, Y_k)\}_{k=0}^{K}$, with $Y_0 = ... = Y_K$
  $\mathcal{S}_{query} = \{(X_q, Y_q)\}_{q=0}^{Q}$
  $R_{0:K} = f_\theta(X_{0:K})$
  $\alpha_{0:K} = \mathit{Softmax}[f_\rho(R_{0:K})]$
  $ME = \sum_{k=0}^{K} [R_k * \alpha_k]$
  $\psi, \phi = \{W_i, \rho\}, \{\theta, W_i, \rho\}$
  $\psi \leftarrow \psi - \alpha \nabla_\psi \ell_i(f_\psi(ME), Y_{0:K})$
  $\phi \leftarrow \phi - \beta \nabla_\phi \ell_i(f_\phi(X_{0:Q}), Y_{0:Q})$
end while

References

[1] Antoniou, A., Patacchiola, M., Ochal, M., Storkey, A.J.: Defining benchmarks for continual few-shot learning. ArXiv **abs/2004.11967** (2020)

[2] Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: European Conference on Computer Vision. (2018)

[3] Asano, Y.M., Rupprecht, C., Vedaldi, A.: Self-labelling via simultaneous clustering and representation learning. In: International Conference on Learning Representations. (2020)

[4] Rebuffi, S.A., Kolesnikov, A.I., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. 2017 IEEE Conference on Computer Vision and Pattern Recognition

(CVPR) (2017) 5533–5542

[5] Rao, D., Visin, F., Rusu, A.A., Teh, Y.W., Pascanu, R., Hadsell, R.: Continual unsupervised representation learning (2019)

[6] Hsu, K., Levine, S., Finn, C.: Unsupervised learning via meta-learning. In: International Conference on Learning Representations. (2019)

[7] Khodadadeh, S., Bölöni, L., Shah, M.: Unsupervised meta-learning for few-shot image and video classification. ArXiv **abs/1811.11819** (2018)

[8] Ji, Z., Zou, X., Huang, T., Wu, S.: Unsupervised few-shot learning via self-supervised training. ArXiv **abs/1912.12178** (2019)

[9] Lee, H.B., Lee, H., Na, D., Kim, S., Park, M., Yang, E., Hwang, S.J.: Learning to balance: Bayesian meta-learning for imbalanced and out-of-distribution tasks. In: International Conference on Learning Representations. (2020)

[10] Lee, K., Lee, K., Shin, J., Lee, H.: Overcoming catastrophic forgetting with unlabeled data in the wild. In: ICCV. (2019)

[11] Javed, K., White, M.: Meta-learning representations for continual learning. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R., eds.: Advances in Neural Information Processing Systems 32. Curran Associates, Inc. (2019) 1818–1828

[12] Beaulieu, S., Frati, L., Miconi, T., Lehman, J., Stanley, K.O., Clune, J., Cheney, N.: Learning to continually learn. 24th European Conference on Artificial Intelligence (ECAI) (2020)

[13] Caccia, M., Rodriguez, P., Ostapenko, O., Normandin, F., Lin, M., Caccia, L., Laradji, I., Rish, I., Lacoste, A., Vazquez, D., et al.: Online fast adaptation and knowledge accumulation: a new approach to continual learning. arXiv preprint arXiv:2003.05856 (2020)

[14] Rajasegaran, J., Khan, S., Hayat, M., Khan, F.S., Shah, M.: itaml : An incremental task-agnostic meta-learning approach. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020)

[15] Perez, E., Strub, F., De Vries, H., Dumoulin, V., Courville, A.: Film: Visual reasoning with a general conditioning layer. arXiv preprint arXiv:1709.07871 (2017)

[16] Zintgraf, L., Shiarlis, K., Kurin, V., Hofmann, K., Whiteson, S.: Fast context adaptation via meta-learning. In: Thirty-sixth International Conference on Machine Learning (ICML). (June 2019)

[17] Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: ICLR. (2017)

[18] Krizhevsky, A.: Learning multiple layers of features from tiny images. Technical report, Canadian Institute for Advanced Research (2009)

[19] Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., Perona, P.: Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology (2010)