NEURAL INFORMATION PROCESSING SYSTEMS

FeCAM: Exploiting the Heterogeneity of Class Distributions in Exemplar-Free Continual Learning

Dipam Goswami, Yuyang Liu, Bartłomiej Twardowski and Joost van de Weijer

Code: https://github.com/dipamgoswami/FeCAM

Computer Vision Center

UAB Universitat Autònoma de Barcelona

IDEAS NCBR

Exemplar-Free Continual Learning:

- ➔ Class-Incremental Learning (CIL): The objective is to learn new classes in a continual fashion without forgetting the knowledge learned from previous tasks.
- ➔ Exemplar-based methods store samples (or exemplars) from previous tasks to use them in training during new tasks. This helps to reduce catastrophic forgetting of old classes.
- ➔ We explore the more challenging *exemplar-free* setting in this work where we do not use any samples from previous tasks.
- ➔ Similar to recent works [1], we train the feature extractor in the first task and then freeze it during the new tasks.

[1] Grégoire Petit, Adrian Popescu, Hugo Schindler, David Picard, and Bertrand Delezoide. Fetril: Feature translation for exemplar-free class-incremental learning. In Winter Conference on Applications of Computer Vision (WACV), 2023.

FeCAM: Feature **C**ovariance-**A**ware **M**etric

- ➔ We explore prototypical networks for CIL, which generate new class prototypes using the frozen feature extractor and classify the features based on the *Euclidean distance* to the prototypes known as NCM (Nearest Class Mean) classifier.
- ➔ We analyze that classification based on Euclidean metrics is successful for jointly trained features.
- ➔ However, when learning from non-stationary data, we observe that the Euclidean metric is suboptimal and that feature distributions are heterogeneous.
- ➔ To address this challenge, we revisit the anisotropic *Mahalanobis distance* for CIL.

Mahalanobis Distance

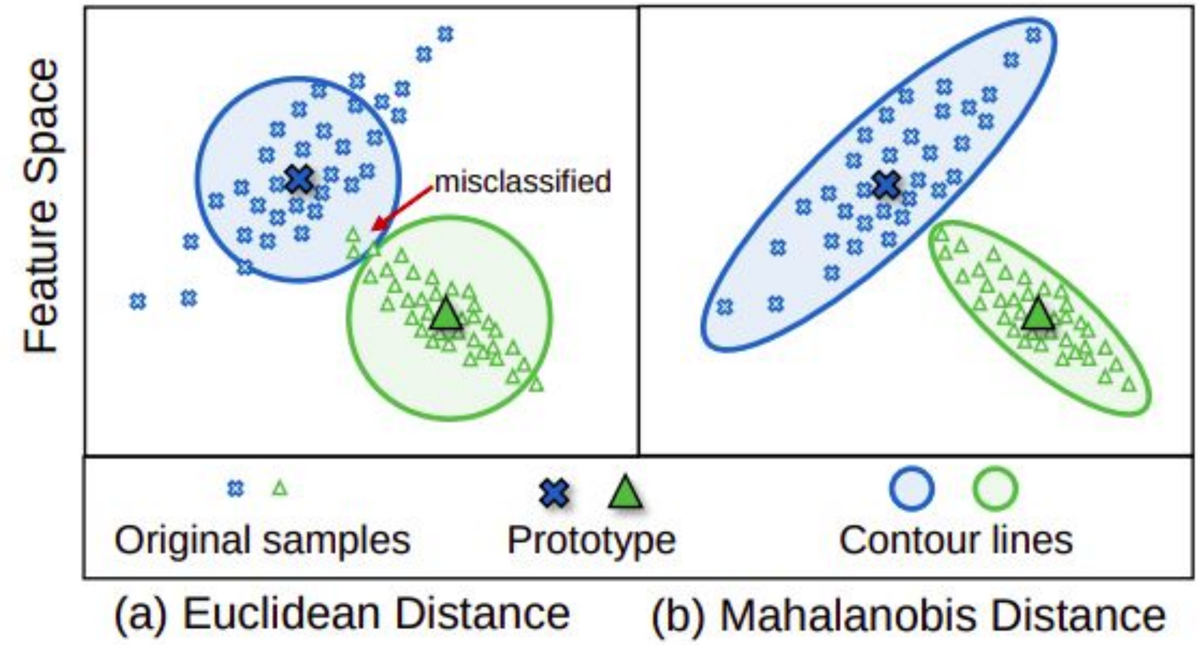
The Mahalanobis distance is generally used to measure the distance between a data sample x and a distribution $\mathcal{D}$. Given the distribution has a mean representation μ and an invertible covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{D \times D}$, then the squared Mahalanobis distance can be expressed as:

$$\mathcal{D}_M(x, \mu) = (x - \mu)^T \mathbf{\Sigma}^{-1} (x - \mu) \tag{1}$$

where $\mathbf{\Sigma}^{-1}$ is the inverse of the covariance matrix.

In euclidean space, $\mathbf{\Sigma} = I$, where I is an identity matrix. Thus, in euclidean space, we consider identical variance along all dimensions and ignore the positive and negative correlations between the variables.

Mahalanobis Distance



Contour lines indicate points at equal distance from the prototype

When did we start using the euclidean metric?

- Before the emergence of deep neural networks, Mahalanobis distance [2] was used in the NCM classifier to assign an image to the class with the closest mean:

$$y^* = \underset{y=1,\ldots,Y}{\operatorname{argmin}} \mathcal{D}_M(x, \mu_y), \quad \mathcal{D}_M(x, \mu_y) = (x - \mu_y)^T M (x - \mu_y) \tag{1}$$

where Y is the number of classes, $x, \mu_y \in \mathbb{R}^D$, class mean $\mu_y = \frac{1}{|X_y|} \sum_{x \in X_y} x$, and M is a positive definite matrix. They learned a low-rank matrix $M = W^T W$ where $W \in \mathbb{R}^{m \times D}$, with $m \leq D$.

[2] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2013.

➔ However, with the shift towards deep feature representations, Guerriero et al. [3] assert that the highly non-linear nature of learned representations with a deep convolutional network eliminate the need of learning the Mahalanobis metric M and the isotropic Euclidean distance can be used as follows:

$$y^* = \operatorname*{argmin}_{y=1,\ldots,Y} \mathcal{D}_e(\phi(x), \mu_y), \quad \mathcal{D}_e(\phi(x), \mu_y) = (\phi(x) - \mu_y)^T (\phi(x) - \mu_y) \tag{2}$$

➔ NCM classifier with euclidean distance is commonly used in continual learning following iCaRL [4].

[3] Samantha Guerriero, Barbara Caputo, and Thomas Mensink. Deepncm: Deep nearest class mean classifiers. International Conference on Learning Representations Workshop (ICLR-W), 2018.
[4] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

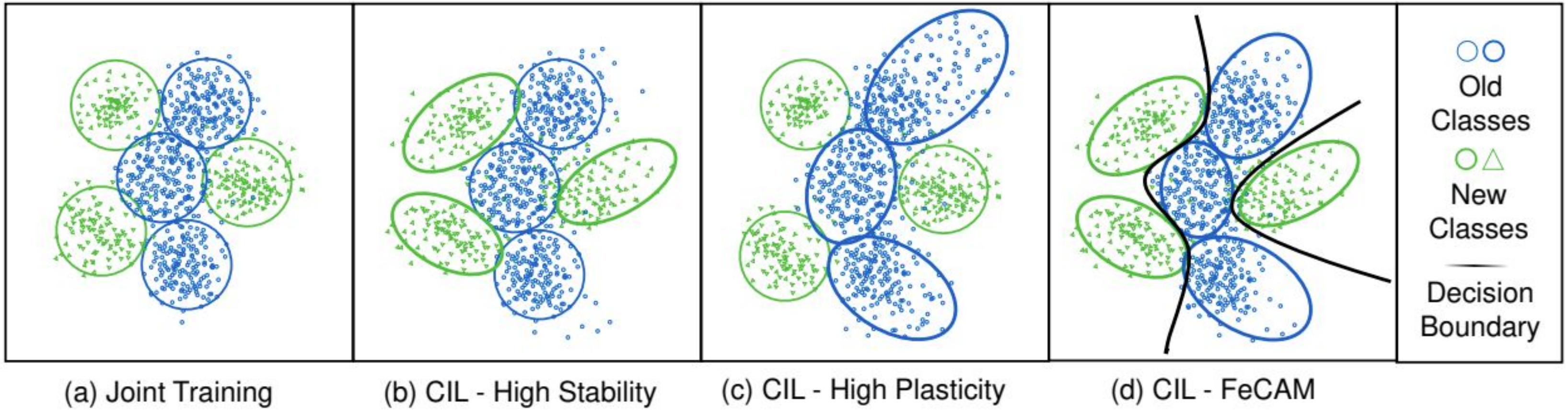
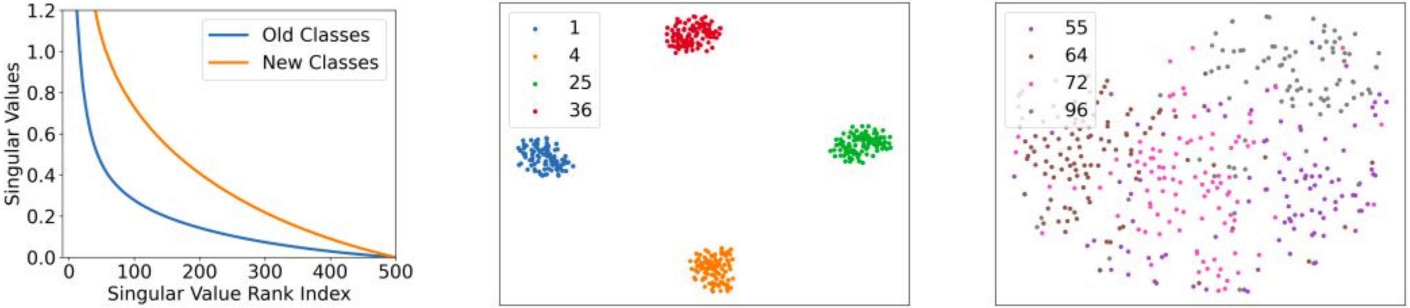


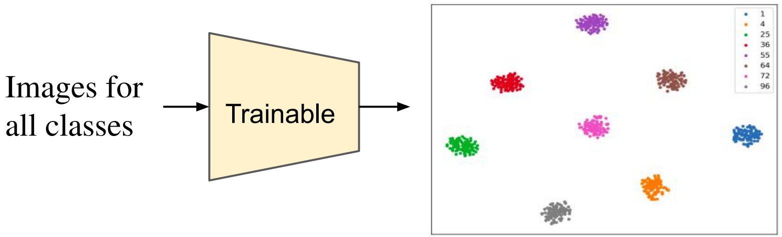
Figure 1: Illustration of feature representations in CIL settings. In Joint Training (a), deep neural networks learn good isotropic spherical representations [16] and thus the Euclidean metric can be used effectively. However, it is challenging to learn isotropic representations of both old and new classes in CIL settings. When the model is too stable in (b), it is unable to learn good spherical representations of new classes and when it is too plastic in (c), it learns spherical representations of new classes but loses the spherical representations of old classes. Thus, it is suboptimal to use the isotropic euclidean distance. We propose FeCAM in (d) which models the feature covariance relations using Mahalanobis metric and learns better non-linear decision boundaries for new classes.

Analysis of feature Distributions

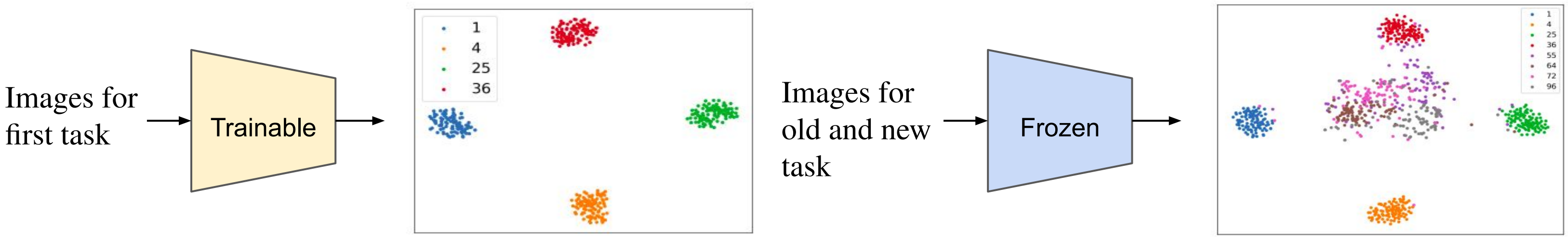


(a) Singular values (b) Old classes (c) New classes

Figure 3: (a) Singular values comparison for old and new classes, (b-c) Visualization of features for old classes and new classes by t-SNE, where the colors of points indicate the corresponding classes.



<u>Joint Training</u>: Isotropic euclidean metric makes sense [2]



First Task

Second Task

[3] Samantha Guerriero, Barbara Caputo, and Thomas Mensink. Deepncm: Deep nearest class mean classifiers. International Conference on Learning Representations Workshop (ICLR-W), 2018.

Covariance Shrinkage

- ➔ When the number of samples available for a class is less than the number of feature dimensions, we obtain a low-rank matrix and the covariance matrix Σ is not invertible.
- ➔ This is a serious problem since the feature dimensions are very high (512 or 768).
- ➔ In order to obtain a full-rank invertible covariance matrix, we perform covariance shrinkage.

$$\boldsymbol{\Sigma}_s = \boldsymbol{\Sigma} + \gamma_1 V_1 I + \gamma_2 V_2 (1 - I), \tag{8}$$

where V_1 is the average diagonal variance, V_2 is the average off-diagonal covariance of $\boldsymbol{\Sigma}$ and I is an identity matrix.

Correlation Normalization of Covariance Matrices

- ➔ The covariance matrix obtained for each class will have different levels of scaling and variances along different dimensions.
- ➔ Particularly, due to the notable shift in feature distributions between the old and new classes, the variances are much higher for the new classes.
- ➔ In order to make the multiple covariance matrices comparable, we perform a correlation matrix normalization on all the covariance matrices make their diagonal elements equal to 1.

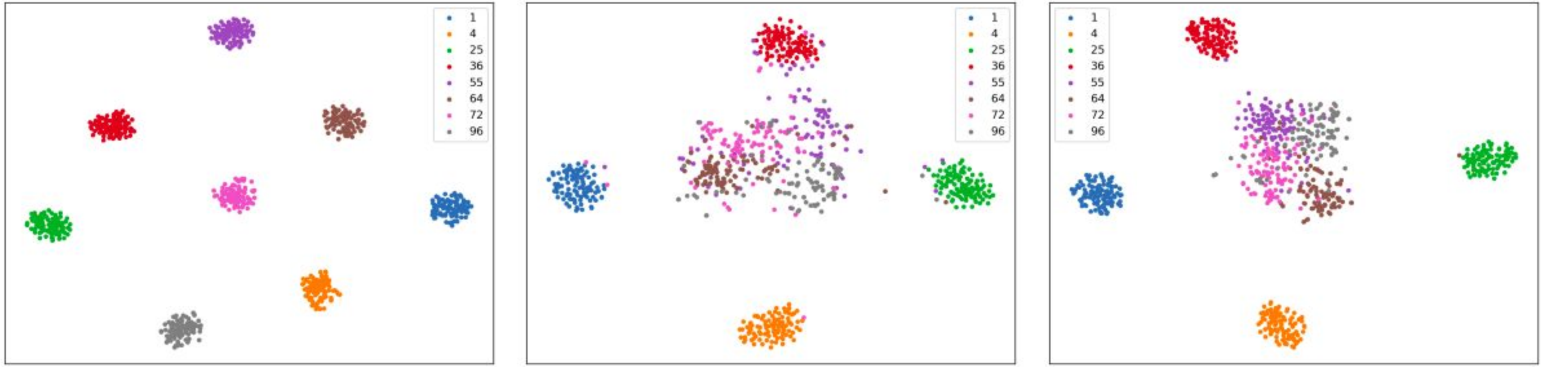
$$\hat{\mathbf{\Sigma}}_y(i,j) = \frac{\mathbf{\Sigma}_y(i,j)}{\sigma_y(i)\sigma_y(j)}, \quad \sigma_y(i) = \sqrt{\mathbf{\Sigma}_y(i,i)}, \quad \sigma_y(j) = \sqrt{\mathbf{\Sigma}_y(j,j)} \tag{7}$$

where $\sigma_y(i)$ and $\sigma_y(j)$ refers to the standard deviations along the dimensions i and j respectively.

Tukey’s Ladder of Powers Transformation

➔ To reduce the skewness of distributions and make them more Gaussian-like.

$$\tilde{\phi(x)} = \begin{cases} \phi(x)^{\lambda} & \text{if } \lambda \neq 0 \\ \log(\phi(x)) & \text{if } \lambda = 0 \end{cases} \tag{9}$$



(a) Joint-Training (b) CL-Frozen backbone (c) CL-Frozen backbone with Tukeys transformation

Algorithm 1 FeCAM

Require: Training data $(D_1, D_2, .., D_T)$, Test data for evaluation $(X_1^e, X_2^e, .., X_T^e)$, Model ϕ

for task $t \in [1, 2, .., T]$ **do**
  if $t == 1$ **then**
    Train ϕ on $D_1 = (X_1, Y_1)$ ▷ Train the feature extractor
  end if
  for $y \in Y_t$ **do**
    $\mu_y = \frac{1}{|X_y|} \sum_{x \in X_y} \phi(x)$ ▷ Compute the prototypes
    $\phi(\tilde{X}_y) = Tukeys(\phi(X_y))$ ▷ Tukeys transformation Eq. (9)
    $\mathbf{\Sigma}_y = Cov(\phi(\tilde{X}_y))$ ▷ Compute the covariance matrices
    $(\mathbf{\Sigma}_y)_s = Shrinkage(\mathbf{\Sigma}_y)$ ▷ Apply covariance shrinkage Eq. (8)
    $(\hat{\mathbf{\Sigma}}_y)_s = Normalization((\mathbf{\Sigma}_y)_s)$ ▷ Apply correlation normalization Eq. (7)
  end for
  for $x \in X_t^e$ **do**
    $y^* = \underset{y=1,\dots,Y_t}{\operatorname{argmin}} \mathcal{D}_M(\phi(x), \mu_y)$ where
    $\mathcal{D}_M(\phi(x), \mu_y) = (\tilde{\phi(x)} - \tilde{\mu}_y)^T (\hat{\mathbf{\Sigma}}_y)_s^{-1} (\tilde{\phi(x)} - \tilde{\mu}_y)$
    ▷ Compute the squared mahalanobis distance to prototypes
  end for
end for

Many-shot CIL Experiments

CIL Method	CIFAR-100			TinyImageNet			ImageNet-Subset		
	T=5	T=10	T=20	T=5	T=10	T=20	T=5	T=10	T=20
EWC [24]	24.5	21.2	15.9	18.8	15.8	12.4	-	20.4	-
LwF-MC [43]	45.9	27.4	20.1	29.1	23.1	17.4	-	31.2	-
DeeSIL [2]	60.0	50.6	38.1	49.8	43.9	34.1	67.9	60.1	50.5
MUC [31]	49.4	30.2	21.3	32.6	26.6	21.9	-	35.1	-
SDC [67]	56.8	57.0	58.9	-	-	-	-	61.2	-
PASS [78]	63.5	61.8	58.1	49.6	47.3	42.1	64.4	61.8	51.3
IL2A [77]	66.0	60.3	57.9	47.3	44.7	40.0	-	-	-
SSRE [79]	65.9	65.0	61.7	50.4	48.9	48.2	-	67.7	-
FeTrIL* [42]	67.6	66.6	63.5	55.4	54.3	53.0	73.1	71.9	69.1
Eucl-NCM	64.8	64.6	61.5	54.1	53.8	53.6	72.2	72.0	68.4
FeCAM (ours) - $\mathbf{\Sigma}^{1:t}$	$\underline{68.8}$	$\underline{68.6}$	$\underline{67.4}$	$\underline{56.0}$	$\underline{55.7}$	$\underline{55.5}$	$\underline{75.8}$	$\underline{75.6}$	$\underline{73.5}$
FeCAM (ours) - $\mathbf{\Sigma}_y$	**70.9**	**70.8**	**69.4**	**59.6**	**59.4**	**59.3**	**78.3**	**78.2**	**75.1**
Upper Bound	79.2	79.2	79.2	66.1	66.1	66.1	81.2	81.2	81.2

Comparison with exemplar-based methods

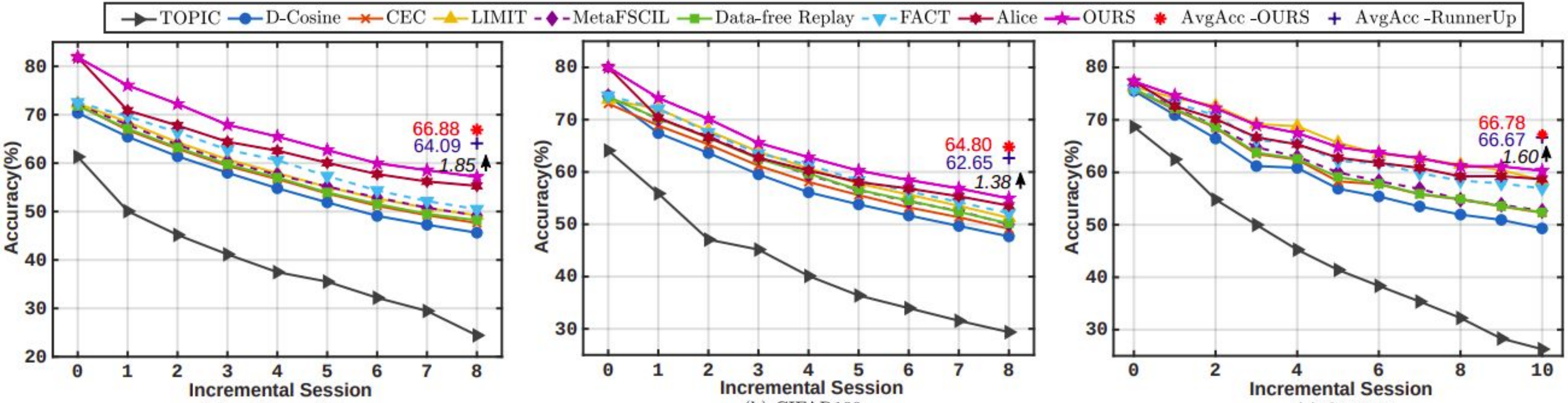
CIL Method	#P	Ex.	CIFAR-100 (T = 5) Avg. $\mathcal{A}$cc	CIFAR-100 (T = 5) Last $\mathcal{A}$cc	ImageNet-Subset (T = 5) Avg. $\mathcal{A}$cc	ImageNet-Subset (T = 5) Last $\mathcal{A}$cc
iCaRL [43]	11.17	✓	65.4	56.3	62.6	53.7
PODNet [15]	11.17	✓	67.8	57.6	73.8	62.9
Coil [76]	11.17	✓	-	-	59.8	43.4
WA [70]	11.17	✓	69.9	61.5	65.8	56.6
BiC [63]	11.17	✓	66.1	55.3	66.4	49.9
FOSTER [59]	11.17	✓	67.9	60.2	69.9	63.1
DER [65]	67.02	✓	**73.2**	**66.2**	<u>77.6</u>	**71.1**
MEMO [74]	53.14	✓	-	-	76.7	70.2
FeCAM(ours)	11.17	✗	<u>70.9</u>	<u>62.1</u>	**78.3**	<u>70.9</u>

NEURAL INFORMATION PROCESSING SYSTEMS

Experiments with ViT models pre-trained on ImageNet-21k

CIL Method	Split-Cifar100 Avg $\mathcal{A}cc$	Split-ImageNet-R Avg $\mathcal{A}cc$	CoRe50 Test $\mathcal{A}cc$
FT-frozen	17.7	39.5	-
FT	33.6	28.9	-
EWC [24]	47.0	35.0	74.8
LwF [29]	60.7	38.5	75.5
L2P [62]	83.8	61.6	78.3
NCM [21]	83.7	55.7	85.4
FeCAM (ours)	**85.7**	**63.7**	**89.9**
Joint	90.9	79.1	-

Accuracy of each incremental task for Few-shot CIL



(a) miniImageNet (b) CIFAR100 (c) CUB200

Ablation experiments

Distance	Cov. Matrix	Tukey Eq. (9)	Shrinkage Eq. (8)	Norm. Eq. (7)	CIFAR-100 (T=5)		ImageNet-Subset (T=5)	
					Last Acc	Avg Acc	Last Acc	Avg Acc
Euclidean	-	✗	-	-	51.6	64.8	60.0	72.2
Euclidean	-	✓	-	-	54.4	66.6	66.2	73.6
Mahalanobis	Full	✗	✗	✗	14.6	29.7	33.5	45.1
Mahalanobis	Full	✓	✗	✗	20.6	36.2	54.0	65.6
Mahalanobis	Full	✗	✓	✗	44.6	59.3	39.9	56.9
Mahalanobis	Full	✓	✓	✗	52.1	62.8	56.5	67.3
Mahalanobis	Diagonal	✓	✓	✗	55.2	66.9	64.0	74.1
Mahalanobis	Full	✗	✓	✓	55.4	65.9	58.1	68.5
Mahalanobis	Full	✓	✓	✓	**62.1**	**70.9**	**70.9**	**78.3**

- ➔ Time complexity on ImageNet-Subset using one Nvidia RTX 6000 for 5 new tasks:
  - ◆ The fastest method FeTrIL takes 44 minutes for all the new tasks while FeCAM takes only 6 minutes with no training.

Thanks