Appendix: Unseen Classes at a Later Time? No Problem

This appendix provides additional details that could not be included in the main manuscript owing to space constraints. In particular, we provide:

- Details of datasets chosen for studies.
- Details of task-wise data splits for each of the three problem settings: Static, Dynamic, Online
- Metrics used for evaluating our model.
- Qualitative results depicting the similarity scores of visual features across tasks.
- Analysis of the generative module
  - t-SNE visualizations to show contribution of various components of our approach and comparison with sequentially trained generative ZSL baseline.
  - Performance of the model on varying the number of replayed samples.
- Implementation details of our method.

Code for all experiments can be accessed at : https://github.com/sumitramalagi/Unseen-classes-at-a-later-time

A1. Dataset Details

In this section, we provide a detailed description of the benchmark datasets used for evaluating our model in the *static*, *dynamic* and *online* CGZSL settings. Following the existing literature [6–9, 26] we assess our proposed model on five widely used benchmark datasets i.e AWA1, AWA2, CUB, SUN, aPY, which are traditionally used for zero-shot learning. (Zero-shot recognition datasets are used since we require access to semantic attributes for addressing the CGZSL problem [6–9, 26])
The Animals with Attributes dataset (AWA1 and AWA2) [11] consists of 50 classes of animals captured in diverse backgrounds. AWA1 consists of 30,475 images and AWA2 consists of 37,322 images. They are split into 40 seen classes and 10 unseen classes. The dataset also contains an 85-dimensional attribute vector for each class which is annotated by a human. Caltech UCSD Birds 200 (CUB) [28] dataset consists of 11,788 images of birds in total, each of which belongs to one of the 200 classes. In a standard generalized zero-shot learning (GZSL) setup, 150 of these classes are treated as seen and 50 classes are unseen. Each class in CUB has nearly 60 samples. In the CUB dataset, each class is also provided with a 312-dimensional human-annotated class attribute vector. The scene recognition dataset (SUN) [21] consists of 717 scenes or classes. Out of 717 classes, 645 classes are seen and the rest 72 are unseen. This dataset contains 14,340 fine-grained images and each class is associated with a 102-dimensional attribute. In aPY [3] dataset, there are 15,339 total images belonging to 32 classes. 20 of these classes are treated as seen and 12 are unseen. Each class is associated with a 64-dimensional attribute. We summarize the details of all datasets in Table A1.

Dataset	Attribute	# Images	Seen Class	Unseen Class
AWA1	85	30,475	40	10
AWA2	85	37,322	40	10
aPY	64	15,339	20	12
CUB	312	11,788	150	50
SUN	102	14,340	645	72

Table A1. Details of datasets used for zero-shot learning

Following protocol in [6, 7, 9, 27, 32], the visual features for all datasets are extracted using ResNet-101 pretrained on Imagenet dataset. We use the publicly available version of benchmark datasets provided by [32].

A2. Task-wise Data Splits

Unlike traditional GZSL methods where all classes are available during training/testing, continual GZSL (CGZSL) settings work on incremental tasks. As described in Sec. 3 of the main manuscript, the pattern in which new classes arrive in CGZSL depends on the setting (static, dynamic, online). Details of the task-wise split for standard zero-shot learning datasets with respect to various settings is described below:

Static CGZSL: For the static CGZSL setting, we follow the dataset split mentioned in [6, 8, 9]. For a given task T_t, the first t subsets i.e data belonging to the current and previous tasks are considered as seen while the rest are unseen. We divide AWA1 and AWA2 datasets into 5 tasks. The first task consists of 10 seen classes and 40 unseen classes. In

	Task-1		Task-2		Task-3		**Task-4 and more**	
	Seen (Replayed+New)	Unseen	Seen(Replayed+New)	Unseen	Seen(Replayed+New)	Unseen	**...**	**...**
			aPY					
Static	0+8	24	8+8	16	16+8	8	**...**	**...**
Dynamic	0+5	3	5+5	6	10+5	9	**...**	**...**
Online	0+4	4	4+5	7	9+5	10	**...**	**...**
			AWA-1					
Static	0+10	40	10+10	30	20+10	20	**...**	**...**
Dynamic	0+8	2	8+8	4	16+8	6	**...**	**...**
Online	0+7	3	7+8	5	15+8	7	**...**	**...**
			AWA-2					
Static	0+10	40	10+10	30	20+10	20	**...**	**...**
Dynamic	0+8	2	8+8	4	16+8	6	**...**	**...**
Online	0+7	3	7+8	5	15+8	7	**...**	**...**
			CUB					
Static	0+10	190	10+10	180	20+10	170	**...**	**...**
Dynamic	0+7	2	7+7	4	14+7	6	**...**	**...**
Online	0+6	3	6+7	5	13+7	7	**...**	**...**
			SUN					
Static	0+47	670	47+47	623	94+47	576	**...**	**...**
Dynamic	0+43	4	43+43	8	86+43	12	**...**	**...**
Online	0+42	5	42+43	9	85+43	13	**...**	**...**

Table A2. Data-splits across all datasets for Static, Dynamic and Online settings. During each task, seen classes is the combination of replayed classes from previous task and newly added seen classes.

each subsequent task we convert 10 of the unseen classes to seen. At the end of fifth task all the 50 classes of AWA1 and AWA2 dataset are converted to seen. SUN dataset is divided into 15 tasks with 47 unseen classes getting converted to seen in each task. CUB dataset is divided into 20 tasks where we incrementally convert 10 unseen classes into seen. aPY is split into 4 tasks, with each new task 8 previously unseen classes are converted to seen class.

Dynamic CGZSL: In the dynamic CGZSL setting, new seen and unseen classes are added in each task. AWA1 and AWA2 datasets are divided into five tasks. In each task, 8 new seen and 2 new unseen classes are added. SUN dataset is divided into 15 task, where 43 seen classes and 4 unseen classes are added in each task. CUB dataset is divided into 20 tasks, where 7 seen classes and 2 unseen classes are added in each task. The aPY dataset consists of four tasks, with 5 seen classes and 3 unseen classes in each task.

Online CGZSL: In our proposed online-CGZSL setting, each task has a disjoint set of seen and unseen classes. In addition, some of the previously unseen classes can turn into seen if the corresponding visual features become available for training in future tasks. To evaluate our model we consider the case where one of the previously unseen class is converted to seen class. AWA1 and AWA2 datasets are divided into five tasks. Every task consists of seven seen classes and three unseen classes. In addition, for each of task numbers two to five, one of the previously unseen classes is converted to a seen class. SUN dataset is divided into 15 tasks with 42 seen and 5 unseen classes. 6 seen and 3 unseen classes are added in each task for the CUB dataset over 20 tasks. aPY dataset is divided into four tasks with four seen and unseen classes. Similar to AWA1 and AWA2, one of the previously unseen classes is converted to a seen class during each task for CUB, SUN and aPY datasets. The data-splits for all settings are listed in the Table A2.

A3. Evaluation Metrics

Static setting: We follow the evaluation metrics mentioned in [26]:

- Mean Seen Accuracy (mSA)

$$\frac{1}{T}\sum_{t=1}^{T} CAcc(D_{te}^{\leq t}, A^{\leq t}) \tag{A1}$$

- Mean Unseen Accuracy(mUA)

$$\frac{1}{T-1}\sum_{t=1}^{T-1} CAcc(D_{te}^{>t}, A^{>t}) \tag{A2}$$

- Mean Harmonic Accuracy (mH)

$$\frac{1}{T-1}\sum_{t=1}^{T-1} H(D_{te}^{\leq t}, D_{te}^{>t}, A) \tag{A3}$$

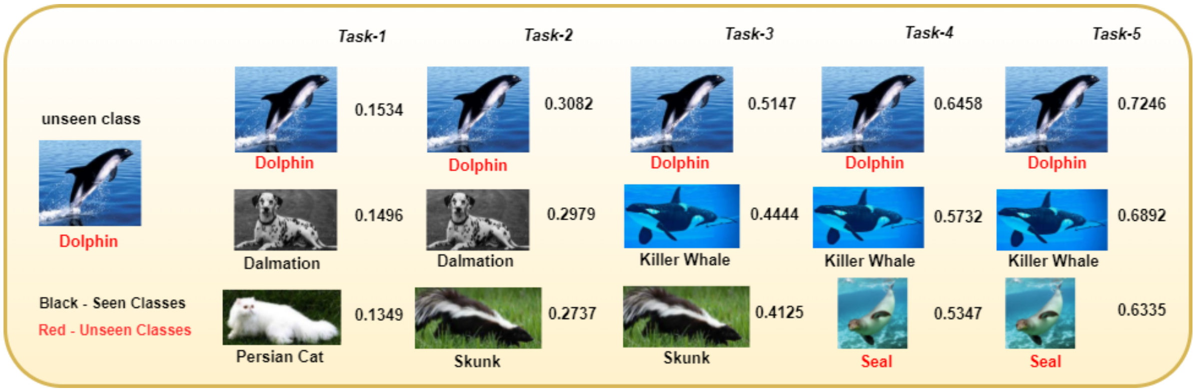

Figure A1. Cosine similarity scores of unseen class '*dolphin*' of AWA2 dataset w.r.t identifier projection. Top 3 cosine similarity scores shared with '*dolphin*' at every task is shown. Unseen classes are depicted in red, seen classes are in black.

Here, $CAcc$ stands for per class accuracy, H represents harmonic mean, T denotes the total number of tasks, $D_{te}^{\leq t}$ denotes test data till t^{th} task,which according to setting-1, corresponds to seen data and $D_{te}^{>t}$ denotes test data of future tasks with respect to t^{th} task. As per static setting, future task data is the unseen data. A denotes the set of all attributes.

Dynamic setting: We use a similar evaluation metrics as mentioned in [6, 7] :

- Mean Seen Accuracy (mSA)

$$\frac{1}{T}\sum_{t=1}^{T} CAcc(D_{te_s}^{\leq t}, A^{\leq t}) \tag{A4}$$

- Mean Unseen Accuracy(mUA)

$$\frac{1}{T}\sum_{t=1}^{T} CAcc(D_{te_u}^{\leq t}, A^{\leq t}) \tag{A5}$$

- Mean Harmonic Accuracy (mH)

$$\frac{1}{T}\sum_{t=1}^{T} H(D_{te_s}^{\leq t}, D_{te_u}^{\leq t}, A^{\leq t}) \tag{A6}$$

Here, $CAcc$ stands for per class accuracy, H represents harmonic mean, T denotes the total number of tasks, $D_{te_s}^{\leq t}$ denotes test data of seen classes till t^{th} and $D_{te_u}^{\leq t}$ denotes test data of unseen classes till t^{th} task. $A^{\leq t}$ denotes the set of all attributes encountered so far.

Online setting: We use the evaluation metrics proposed in dynamic setting considering the updated set of seen and unseen classes for calculating accuracy.

We re-calculate each task's accuracy in order to obtain the average accuracy for a given task.

A3.1. Additional Evaluation Metrics

mAUSUC: [26] adopted the mean area under seen/unseen curve (mAUSUC) as metric for measuring the performance of CGZSL models across all tasks. mAUSUC is given by:

$$mAUSUC(F) = \frac{1}{T}\sum_{t=1}^{T} AUSUC(F, D_{te}^{\leq t}, A^{\leq t}) \tag{A7}$$

where T is the total number of tasks encountered so far, $D_{te}^{\leq t}$ is the test data consisting of both seen and unseen classes and $A^{\leq t}$ is the set of attributes encountered so far. We compare our mAUSUC score with recent state-of-the-art approaches such as NM-ZSL [26], Tf-GCZSL [7] and A-CZSL [8]. Figure A7 shows task-wise mAUSUC on AWA2 dataset in all three settings. A higher value of mAUSUC indicates that the model is better able to handle bias between seen and unseen classes. We observe that our model's performance is superior to the existing state-of the art CGZSL methods.

A4. Task-wise Similarity Scores of Visual Features

In the CGZSL settings, seen/unseen classes are added incrementally. Our model retains the previously learned knowledge using incremental bi-directional alignment (Sec. 4.3) and generative replay of visual features from the previous classes (Sec. 4.4). The incremental bi-directional alignment loss is a combination of nuclear loss and semantic alignment loss. Nuclear loss helps in aligning identifier projections in accordance with the real visual features and semantic alignment loss aids in strengthening semantic relationships and knowledge transfer as the visual space evolves and new classes are added over time.

In this section, we analyze how applying semantic alignment mechanism incrementally helps in generating better visual features by strengthening semantic relationships as

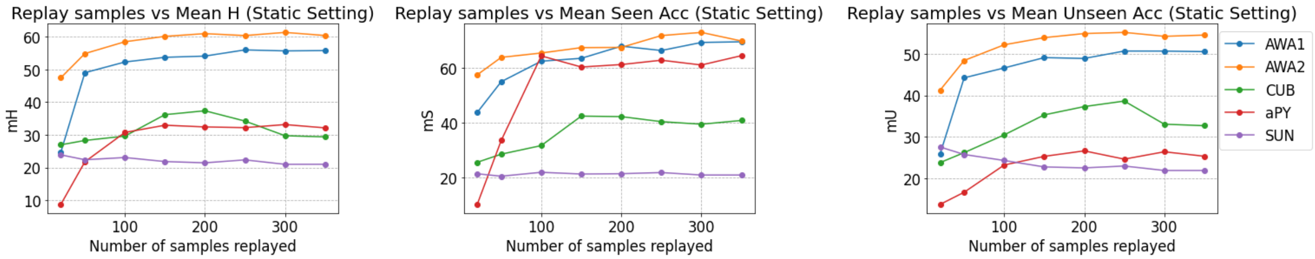

Figure A2. Number of replayed samples vs Mean Harmonic accuracy (mH), Mean Seen accuracy (mS) and Mean Unseen accuracy (mU) on all datasets in Static-CGZSL setting. We observe that fine-grained datasets like CUB and SUN perform well when the number of replayed samples is less. Performance of coarse-grained datasets like AWA1 and AWA2, saturates when the number of replayed samples is more than two hundred.

new classes are added. Since real visual features for unseen classes are not available during training, the semantic alignment loss L_{sal}, tries to leverage semantic structure of all the classes encountered so far and generate accurate unseen features. The seen normalized loss L_{snl} discussed in Sec. 4.2, helps the discriminator to place the identifier projection of unseen classes close to the generated unseen features. Furthermore, the semantic alignment loss is applied with respect to n_c nearest neighbours of class c. As the class distribution changes with time due to dynamic addition of new classes, the visual space evolves and the nearest neighbours for a particular unseen class changes. As new classes that are semantically similar to a given unseen class arrive, performing semantic alignment with nearest classes incrementally helps in further enhancing semantic relationships w.r.t new class distribution and improving the quality of generated unseen class features.

Figure A1 shows cosine similarities calculated during inference between the unseen class '*dolphin*' of the AWA2 dataset and the most closely related seen-unseen classes (in terms of identifier projections) across different tasks. We observe that the alignment of the identifier projections with semantically similar classes during each task. While the model predicts '*dolphin*' correctly across the 5 tasks, during task-1, with the available seen and unseen classes, '*dolphin*' shares a cosine similarity score of 0.1349 with the identifier projection of the unseen class '*persian cat*'. With the addition of new seen and unseen classes during task-2, '*dolphin*' now shares a cosine similarity of 0.2737 with '*skunk*' rather than '*persian cat*' whose cosine similarity dropped to 0.1024 . This distinctly shows that the semantic alignment loss is helping to improve the representation of generated unseen features. The generated unseen features guide discriminator in mapping identifier projections, which in turn aid in classification. It can be seen that at the end of task-5, '*dolphin*' shares high cosine similarity scores with identifier projection of classes '*killer whale*' and '*seal*' which are visually close. This shows how the model leverages current semantic structure to learn better identifier projections as new classes are added.

More such examples are shown in Figure 4 presented in the main manuscript (which we explain here owing to space constraints) and Figure A5. In Figure 4, for the unseen class '*sheep*' of the AWA2 dataset encountered during task-1, the similarities with identifier projection of '*killer whale*' and '*skunk*' classes are 0.2765 and 0.2533. With the addition of new seen and unseen classes during task-2, '*sheep*' now shares a cosine similarity of 0.4380 with '*gorilla*' rather than '*killer whale*' whose cosine similarity dropped to 0.2543 . Note however that the '*sheep*' class is classified correctly in all tasks. Figure A5 illustrates an example from the aPY dataset, where the unseen class '*motorbike*' shares high cosine similarity score with the identifier projection of bicycle class. Since '*bicycle*' and '*motorbike*' are visually very similar, '*bicycle*' has the highest cosine similarity with '*motorbike*' in spite of new classes being added. We can observe that cosine similarity between visually related samples keeps increasing as new tasks arrive, ensuring better alignment between the identifier projections.

A5. Analysis of Performance of Generative Replay Module

t-SNE plot visualization: Our model uses generative replay to overcome catastrophic forgetting. We visualize the generated visual features of AWA1 dataset per task (Figure A6) and observe that generated visual features form well-defined clusters. The features belonging to same class are grouped together and far from other classes, this signifies that the generator is able to generate discriminative visual features. Since f-CLSWGAN [32] is a GAN-based approach for solving GZSL problems, we compare the visual features generated by our model with the sequential version of f-CLSWGAN (Seq-fCLSWGAN) [32]. We notice that well-defined clusters are formed during task-1 of Seq-fCLSWGAN training, but subsequently it tends to forget the acquired knowledge and the newly generated features tend

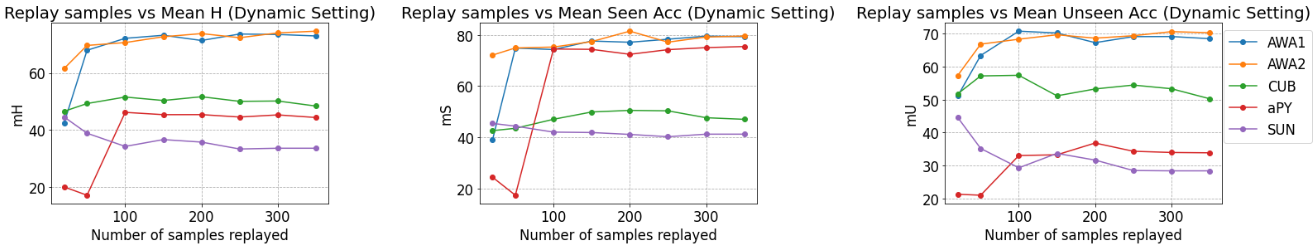

Figure A3. Number of replayed samples vs Mean Harmonic accuracy (mH), Mean Seen accuracy (mS) and Mean Unseen accuracy (mU) on all datasets in Dynamic-CGZSL setting. We observe that performance of coarse-grained datasets like AWA1 and AWA2 increases, if the number of replayed samples is more than hundred.

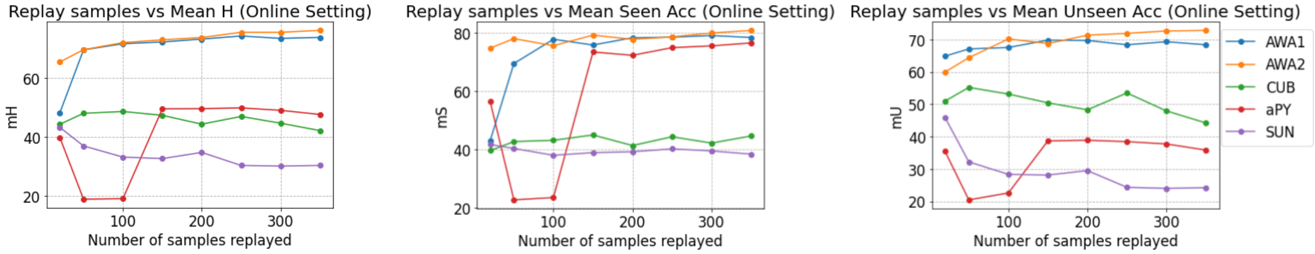

Figure A4. Number of replayed samples vs Mean Harmonic accuracy (mH), Mean Seen accuracy (mS) and Mean Unseen accuracy (mU) on all datasets in Online-CGZSL setting. For fine-grained datasets like SUN, we can notice that replaying lesser samples helps to boost the performance as the number of samples per class are less in SUN dataset. Coarse-grained datasets like AWA1, AWA2 and aPY perform well when the number of replayed samples are more.

to get mixed up in the visual space. We perform ablation study on the proposed approach and show that removing incremental bi-directional alignment and classification loss deforms the clusters across all tasks.

Varying number of replayed samples: We evaluated the performance of our model by varying the number of samples replayed. We compare the number of samples with mean harmonic accuracy (mH), mean seen accuracy (mS) and mean unseen accuracy (mU) in all the three settings and all the datasets. We plot the results in Figure A2 (static) , A3 (dynamic) and A4 (online). We observe that across all settings AWA1, AWA2 perform well when the number of replayed samples is more than 100. We use a replay of 300 for AWA1, AWA2 dataset in our main manuscript. aPY dataset has only 32 classes in total, hence to avoid overfitting we replay only 150 samples per class.

SUN and CUB are fine-grained datasets, and each class has limited data which makes these datasets challenging. In order to mimic the original dataset, we replay only 150 samples per task for CUB and only 20 samples per task for the SUN dataset.

A6. Implementation Details

Our model is implemented using Pytorch-1.4.0 and CUDA-11.2.

The proposed approach consists of a generator G and discriminator D.

We use Adam optimizer with a learning rate of 0.005 and weight_decay of 0.00001 for all settings and datasets. We normalize both target image and attributes before calculating cosine similarity. The total G loss is given by: L_G^t = λ_1 L_{GAN} + λ_2 L_{pcl} + λ_4 L_{iba} and the overall D loss is given by: L_D^t = λ_1 L_{GAN} + λ_2 L_{rcl} + λ_3 L_{snl}, where λ_1, λ_2, λ_3, λ_4 are 1.

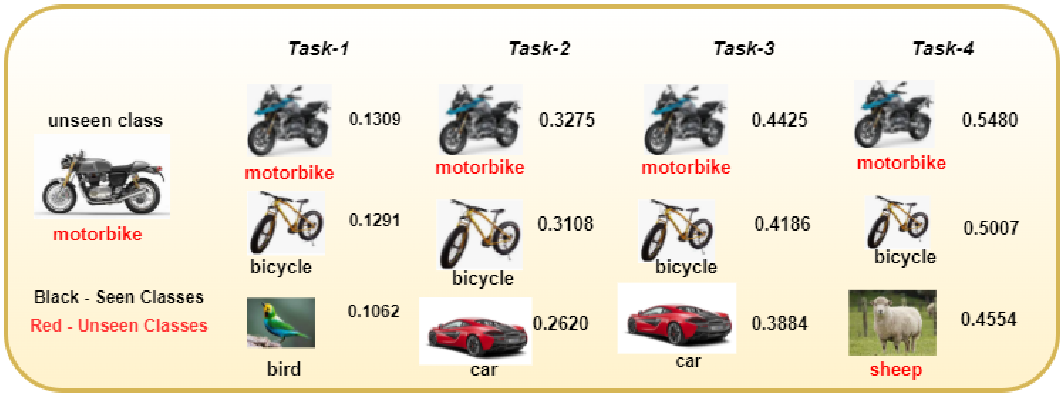

Figure A5. Cosine similarity scores of unseen class '*motorbike*' of aPY dataset. Top three cosine similarity scores shared with '*motorbike*' at every task is shown. Unseen classes are depicted in red, seen classes are in black.

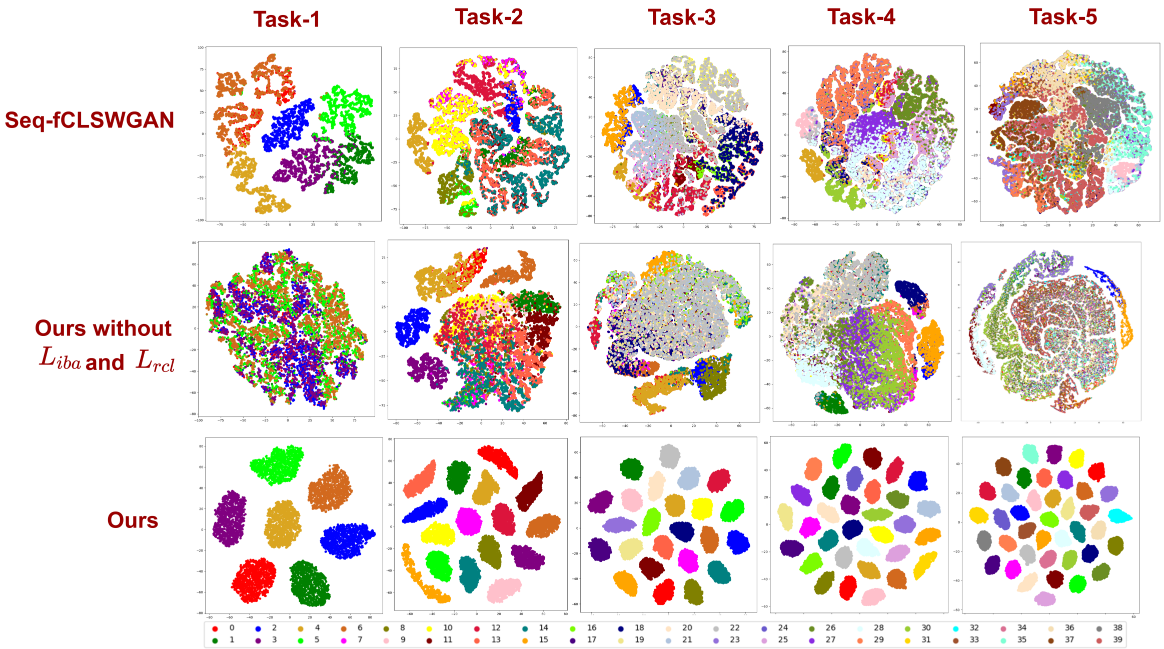

Figure A6. t-SNE visualizations of visual features generated by Seq-fCLSWGAN (Row 1), our method without incremental bi-directional alignment (L_{iba}) and real classification loss (L_{rcl}) (Row 2) and our overall approach (Row 3) during various tasks for AWA1 dataset. Different colors depict different seen classes.

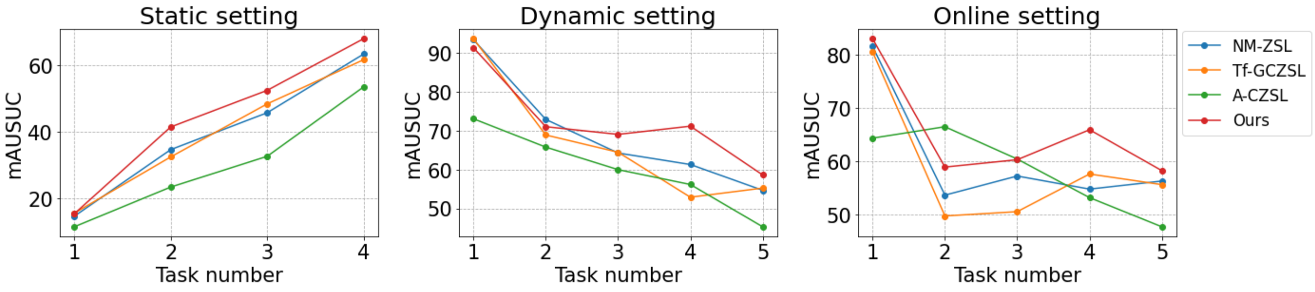

Figure A7. Task-wise mean AUSUC values for static (left), dynamic (center) and online (right).