

# Appendix

## A EXPERIMENTAL SETTINGS

Tables 5 and 6 summarize the hyperparameters used for SOM and VAE across different datasets. In all setups, SOM unit vectors were initialized using random samples from the input data or latent space of the first class seen during training (i.e. class 0), in line with the CIL setup. We also experimented with other initialization strategies such as uniform random initialization over the fixed range of [0,1], and initialization with the global mean of input data, but these methods consistently underperformed compared to the random sampling method. For each SOM unit, the respective statistics were initialized as follows: the mean vector was set to zero, the variance vector to ones and the covariance matrix to zeroes. We also explored other initialization methods for the statistics such as using diagonal covariance matrices, fixed weight initializations and statistics derived from latent vectors from VAE (when VAE is used). However, these alternatives performed less accurately than the basic initialization of zero mean, unit variance and zero covariance. In all experiments, exponential moving averages were maintained for the mean, variance, and covariance of SOM unit activations. The following hyperparameter settings were explored for the bias correction:

Table 4: Bias-correction hyperparameters with best-performing values and additional settings tested.

Parameter	Symbol	Best value	Other values tested
Mean	$\beta_\mu$	0.99	0.98, 0.95
Variance / Cov.	$\beta_\sigma, \beta_\Sigma$	0.95	0.91, 0.99, 0.90

To identify the BMUs in the SOMs, Euclidean distance between the input (latent) vector and the SOM unit weights was used. Euclidean distance produced the most accurate results across the datasets. While we experimented with alternative distance metrics including cosine similarity, Manhattan (L1) distance and Mahalanobis distance (Aly 2014), these methods did not yield any significant improvements in the experiments.

The VAE was trained using the Adam optimizer, which provided consistent reconstruction quality across all datasets. We also evaluated other optimizers such as Stochastic Gradient Descent (SGD) with momentum and Root Mean Square Propagation (RMSprop), but these optimizers resulted in less stable reconstructions and as a result the SOMs replay quality was affected.

---

### Algorithm 2 Synthetic Sample Generation using Mean and Covariance (CIFAR-10)

---

**Require:** Trained SOM, BMU coordinates  $(i, j)$ , number of samples  $n$ , regularization constant  $\epsilon$

- 1:  $\mu \leftarrow \text{SOM.running\_mean}[i][j]$
- 2:  $\Sigma \leftarrow \text{SOM.running\_cov}[i][j]$
- 3:  $\Sigma \leftarrow \Sigma + \epsilon I$  // Add diagonal regularization
- 4: Compute eigen-decomposition:  $Q\Lambda Q^\top = \Sigma$
- 5:  $\Lambda \leftarrow \max(\Lambda, \epsilon)$  // Clamp eigenvalues
- 6:  $\Sigma_{\text{pd}} \leftarrow Q \cdot \text{diag}(\Lambda) \cdot Q^\top$
- 7: Sample  $\tilde{z} \sim \mathcal{N}(\mu, \Sigma_{\text{pd}})$
- 8: **return**  $n$  samples  $\{\tilde{z}_1, \dots, \tilde{z}_n\}$

---

In the CLIP-based experiments (Table 7), we used the pretrained CLIP ViT-B/32 model with weights from `openai`. Unless otherwise noted, the encoder was kept frozen to retain the pretrained visual representation, though in some experiments we optionally fine-tuned only the last transformer block to adapt to the continual learning setup. All images (CIFAR-10/100 at  $32 \times 32$  and TinyImageNet at  $64 \times 64$ ) were internally upsampled to  $224 \times 224$  to match CLIP’s expected resolution. The pooled CLIP embeddings are 512-dimensional, which were then fed into a lightweight transposed-convolutional decoder that reconstructed images at their dataset-native resolution ( $32 \times 32$  for CIFAR-10/100 and  $64 \times 64$  for TinyImageNet).

Table 5: SOM Hyperparameters for MNIST and Fashion-MNIST

Setting	Value
Datasets	MNIST, Fashion-MNIST
Image Size	$28 \times 28$
SOM Sizes ( $n \times n$ )	10, 12, 20, 30, 35
SOM Epochs	10,20,30,50
Sigma	0.95
Neighborhood Function	Gaussian
Distance Metric	Euclidean
SOM Learning Rate	0.5

Table 6: VAE and SOM Hyperparameters for CIFAR-10, CIFAR-100 and TinyImageNet

Component	Hyperparameter Value
<b>VAE Parameters</b>	
Image Size	$32 \times 32 \times 3$ ; $64 \times 64 \times 3$ for TinyImagenet
Latent Dim. ( $n \times 2 \times 2$ )	32, 64, 128
VAE Epochs	200
Batch Size	128, 64
Learning Rate	$1 \times 10^{-4}$ , $1 \times 10^{-5}$ , $5 \times 10^{-5}$
Optimizer	Adam
KL Loss Scale	1.0
Feature Loss Scale	1.0
Norm Type	BatchNorm
Residual Blocks	1 (bottleneck)
Channel Multiplier	32
Block Widths	[1, 2, 4, 8]
<b>SOM Parameters</b>	
SOM Grid Sizes	25, 30, 35, 40
SOM Epochs	150
SOM Learning Rate	0.5, 0.495, 0.45
Sigma	0.95, 0.94, 0.96
Neighbor Function	Gaussian
Distance Metric	Euclidean
Covariance Regularization	$\lambda=1 \times 10^{-4}$ ;
Replay Samples per BMU	$K=1$ (default; can be increased)

## B EFFECT OF SOM RESOLUTION AND VAE CAPACITY ON INCREMENTAL LEARNING

Table 8 reports the accuracy trends for MNIST and Fashion-MNIST under different SOM configurations. We observe that larger SOM resolutions significantly improve performance, with the test accuracy rising with the size of the grid. In MNIST, the precision increases from 87.95% at  $10 \times 10$  to 95.16% at  $35 \times 35$ , approaching the performance of the offline i.i.d. upper bound (95.82%). In FashionMNIST, the performance improves from 72.79% to 81.91% under the same grid expansion of the SOM. These results demonstrate the model’s capacity to retain and learn new knowledge using only synthetic replay generated from SOM statistics. It should also be noted that the accuracy is achieved without any generative model or memory buffer, relying solely on the internal topology and statistics of the SOM for stable representation learning and replay.

Table 9 present the results for CIFAR-10 and CIFAR-100 under the one-class-at-a-time method using a VAE (FT global) hybrid model. We investigated both the size of the SOM and the latent dimensionality of the VAE. For CIFAR-10, the best configuration ( $32 \times 32 \times 2 \times 2$  latent,  $40 \times 40$  SOM) achieves an accuracy of 54.16%, even surpassing the performance of standard split-CIFAR-10



Table 7: CLIP (ViT-B/32) and SOM Hyperparameters for CIFAR-10, CIFAR-100, and TinyImageNet

Component	Hyperparameter Value
<b>CLIP Encoder / Decoder</b>	
Encoder (model)	CLIP ViT-B/32
Pretrained Weights	openai
Fine-tuning	Frozen (default); optional fine-tuning of last transformer block
Embedding Dim.	512 (pooled embedding)
Input Resolution to CLIP	224×224 (upsampled inside the wrapper)
Decoder Target Size	CIFAR-10/100: $32 \times 32 \times 3$ ; TinyImageNet: $64 \times 64 \times 3$
Decoder Type	Transposed-conv upsampler ( $4 \times 4 \rightarrow$ target size)
Decoder Base Channels	1024 (XL); attention at $16^2$ and $32^2$
Decoder Norm / Activations	GroupNorm(32), SiLU; final $\tanh$ (range $[-1, 1]$ )
<b>Training / Optimization</b>	
Batch Size	128
Epochs per Class	200 (cosine LR schedule with warmup)
Optimizer	Adam ( $\beta_1=0.9$ , $\beta_2=0.999$ )
Learning Rate	$1 \times 10^{-4}$ , $1 \times 10^{-5}$ , $5 \times 10^{-5}$
LR Schedule	Warmup 500 steps $\rightarrow$ cosine decay
<b>SOM Parameters</b>	
SOM Grid Size	25,30,35,40
SOM Iterations per Class	150
Learning Rate	0.5, 0.495, 0.45
Sigma (Neighborhood Width)	0.95, 0.94, 0.96
Neighbor Function	Gaussian
Distance Metric	Euclidean
Covariance Regularization	$\lambda=1 \times 10^{-4}$ ;
Replay Samples per BMU	$K=1$ (default; can be increased)

Table 8: Last Accuracy on MNIST and Fashion-MNIST Datasets (10 tasks)

SOM Size	Epochs	Sigma	MNIST	FMNIST
$10 \times 10$	10	0.95	87.95	72.79
$10 \times 10$	50	0.94	87.97	73.83
$12 \times 12$	20	0.95	90.67	76.24
$20 \times 20$	20	0.95	93.01	79.42
$30 \times 30$	30	0.95	94.28	81.08
$35 \times 35$	50	0.95	<b>95.16</b>	<b>81.91</b>

(53.01%). This demonstrates the robustness of our method to finer-grained tasks. In the CIFAR-100 stream, our model achieves an accuracy of 12.41% with the same configuration.

We further evaluate on our VAE (FT BMU specific) method (Table 10), where a separate VAE distribution is associated with each BMU. On CIFAR-10, this method achieved a peak accuracy of 46.10%, lower than the shared VAE (FT global) approach. This might suggest that dividing the latent space too finely could hurt the generalization in the BMUs. A similar trend is observed on CIFAR-100, where the best accuracy is 12.15%. Even though VAE (FT BMU specific) is more complex, it performs worse than the shared latent distribution model. This suggests that using a single shared representation might be more effective for stable memory replay in high-dimensional data.

We extend our evaluation to TinyImageNet, a very complex benchmark consisting of 200 classes, which we structure into a 10-task stream with 20 classes per task. This setting is considerably more challenging due to both the fine-grained nature of the categories and the larger classification space.

Table 9: SOM+VAE Configuration and Last Accuracy on CIFAR-10 (10 tasks), CIFAR-100 (100 tasks), and TinyImageNet (10 tasks).

Latent Dim	SOM Size	VAE+ SOM Epochs	CIFAR-10	CIFAR-100	TinyImageNet
$32 \times 2 \times 2$	$30 \times 30$	200+150	52.32	12.18	6.88
	$35 \times 35$	200+150	54.06	12.20	7.04
	$40 \times 40$	200+150	<b>54.16</b>	<b>12.41</b>	<b>7.14</b>
$64 \times 2 \times 2$	$25 \times 25$	200+150	45.18	11.83	6.89
	$30 \times 30$	200+150	46.53	11.88	6.57
	$35 \times 35$	200+150	37.72	11.01	6.44
$128 \times 2 \times 2$	$25 \times 25$	200+150	44.21	11.13	6.66
	$30 \times 30$	200+150	41.77	11.10	6.12

Table 10: VAE (FT BMU specific) Configuration and Last Accuracy on CIFAR-10 (10 tasks), CIFAR-100 (100 tasks), and TinyImageNet (10 tasks).

Latent Dim	SOM Size	VAE+ SOM Epochs	CIFAR-10	CIFAR-100	TinyImageNet
$32 \times 2 \times 2$	$30 \times 30$	200+150	44.80	11.90	6.11
	$35 \times 35$	200+150	45.60	11.20	6.21
	$40 \times 40$	200+150	<b>46.10</b>	<b>12.15</b>	<b>6.45</b>
$64 \times 2 \times 2$	$25 \times 25$	200+150	42.55	10.82	6.32
	$30 \times 30$	200+150	41.88	10.63	6.35
	$35 \times 35$	200+150	43.40	11.18	6.11
$128 \times 2 \times 2$	$25 \times 25$	200+150	40.85	11.95	6.21
	$30 \times 30$	200+150	40.10	11.88	5.99

The global VAE (FT global) achieves a peak accuracy of 7.14%, higher than the per-BMU variant at 6.45%. These results further confirm the scalability of the shared VAE (FT global) representation, which maintains more stable replay dynamics than its per-BMU counterpart even under highly complex continual learning conditions.

#### B.0.1 REMARKS:

We find that larger SOM sizes consistently lead to better performance across datasets (e.g.  $35 \times 35 > 30 \times 30 > 25 \times 25$ ), which aligns with trends observed on all datasets. However, increasing the SOM size also leads to longer training times and higher computational complexity, as the number of BMUs (neurons) grows quadratically. Similarly, increasing the latent dimensionality in the VAE beyond  $32 \times 2 \times 2$  does not improve the accuracy; it often degrades it. For example, using  $128 \times 2 \times 2$  results in a drop in precision to 38.94% on CIFAR-10. Larger latent spaces are more expensive to train and may introduce sparsity in the SOM map, which might make it harder for the SOM to organize meaningful structures. These results highlight that compact, well-structured latent representations and moderate SOM sizes strike the best balance between performance and efficiency.

**Per-BMU overhead.** Training a separate decoder per unit adds  $\sim 12.6$ M parameters per BMU (Table 11), which is costly at  $40 \times 40$  units. In practice, only BMUs receiving the data are updated, but effective sample counts per BMU are low ( $< 50$  for many units), leading to a poor fit.

## C GENERATIVE CAPABILITY OF THE PROPOSED METHOD

Our proposed model can also be used as a generative model, which is capable of synthesizing samples without storing original data. For the MNIST and FashionMNIST, since we only use the SOM for training, image samples are generated using Gaussian sampling as described in the paper. After training, each SOM unit stores the specific statistics in the form of mean and variance, and they are used to generate sample images. Figure 2 illustrates some of the samples generated for each class using this method.

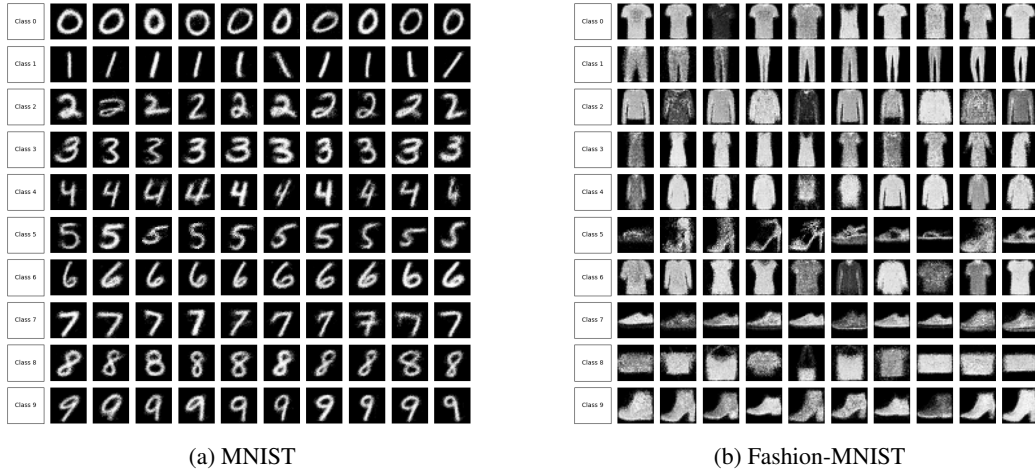


Figure 2: Synthetic samples generated using Gaussian sampling from the trained SOM. Each row corresponds to one class (0–9), and each row contains 10 synthetic samples generated from BMUs labeled with that class.



Figure 3: Synthetic CIFAR-10 samples generated by sampling latent vectors from a Self-Organizing Map (SOM) trained on VAE latent space. Each latent vector was sampled from the full-covariance Gaussian distribution of a Best Matching Unit (BMU) labeled with the target class. The sampled latents were then decoded into images using VAE. Each row corresponds to one class (0–9).

For CIFAR-10 and CIFAR-100, we combine a VAE and the SOM trained on the VAE’s latent space, where, after the training, the SOM statistics store the mean, variance, and full covariance matrix of the latent samples. Synthetic latent samples are then generated using eigen-decomposition of the covariance and then decoded using the VAE decoder to reconstruct the images. Figure 3 and 1 illustrate some of the samples generated for each class using this method.

## D CLASS-LEVEL ANALYSIS (CONFUSION MATRICES)

To complement the aggregate accuracy results presented in the main paper, we include full normalized confusion matrices for all datasets under different variants of our method. Each matrix is row-normalized, allowing us to visualize the distribution of predictions per true class and to highlight which classes are most affected by forgetting or confusion over the course of continual learning. For MNIST and Fashion-MNIST (Figure 4), we evaluate the SOM-only variant without auxiliary generative replay. The resulting matrices exhibit strong diagonal dominance, with MNIST showing nearly perfect separation across all classes. FMNIST, while slightly noisier, retains clear diagonals with only minor confusion among semantically similar clothing items (e.g., shirts vs. coats). These results confirm that SOM-only replay is sufficient for simple, low-variance datasets.

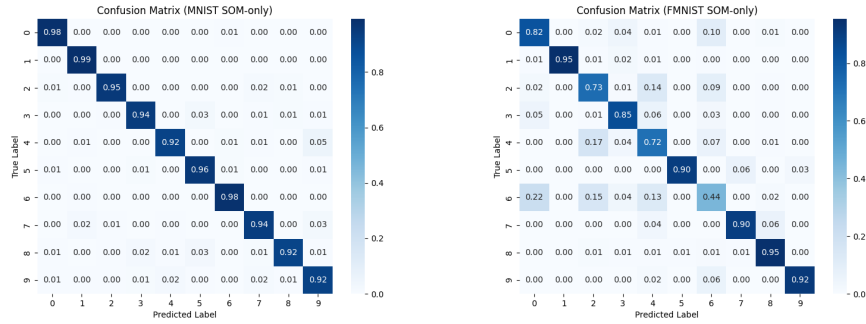


Figure 4: Confusion matrices for MNIST (left) and Fashion-MNIST (right) using the SOM-only variant.

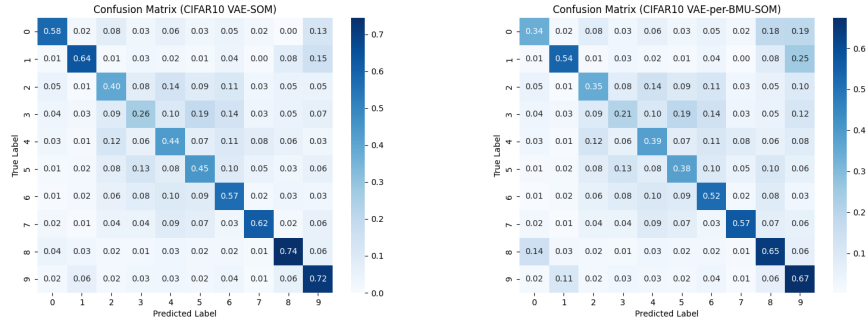


Figure 5: Confusion matrices for CIFAR-10 with VAE (FT global) (left) and VAE (FT BMU specific) (right).

On CIFAR-10 (Figure 5), we compare the VAE (FT BMU specific) and the global VAE (FT global) variants. Both maintain clear diagonal structures, but the per-BMU approach exhibits broader off-diagonal activations, especially among visually similar vehicle categories. By contrast, the global variant produces slightly better diagonals than the global VAE and fewer cross-class confusions, suggesting that aggregating statistics across the full latent space provides a more stable basis for replay.

For CIFAR-100 (Figure 6), where classes are grouped into 10 superclasses, both the per-BMU and global VAE (FT global) variants preserve diagonal structure but also exhibit leakage into neighboring groups. The per-BMU variant shows more dispersion, while the global variant achieves slightly cleaner diagonals and reduced cross-group mixing, suggesting modest but consistent benefits from pooling statistics at the global level. On TinyImageNet (Figure 7), which is the most challenging benchmark with 200 fine-grained classes, both variants again preserve recognizable diagonals but suffer from substantial leakage across adjacent groups due to strong visual overlap between classes. The global VAE (FT global) provides a slight improvement over the per-BMU variant, reinforcing its relative robustness as dataset complexity increases.

Overall, these confusion matrix visualizations reinforce our main results: SOM-based replay maintains per-class retention across diverse benchmarks, and the global VAE (FT global) as well as the per-BMU variant consistently improve stability and scalability as dataset complexity increases.

## E SOM REPRESENTATIONS DURING CONTINUAL LEARNING

Figure 8 visualizes the CIFAR-10 SOM unit vectors after data was fed through the decoders of the VAE and CLIP after the third classes. This highlights a strong feature of this methodology – it is easy to visualize the progress of the algorithm to determine if it is working properly or not, which is

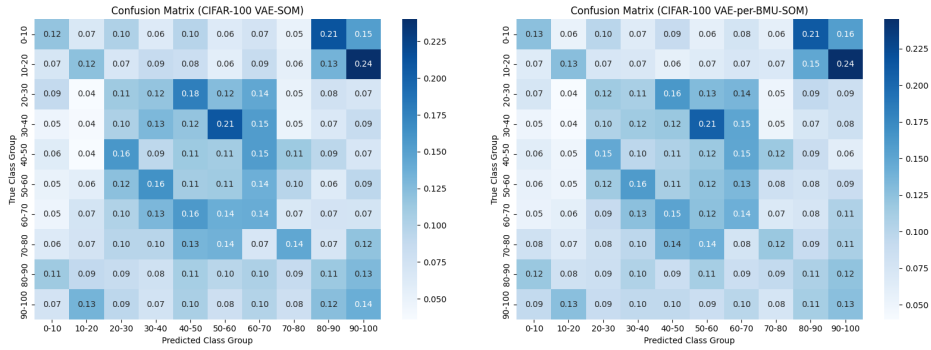


Figure 6: Confusion matrices for CIFAR-100 with VAE (FT global) (left) and VAE (FT BMU specific) (right).

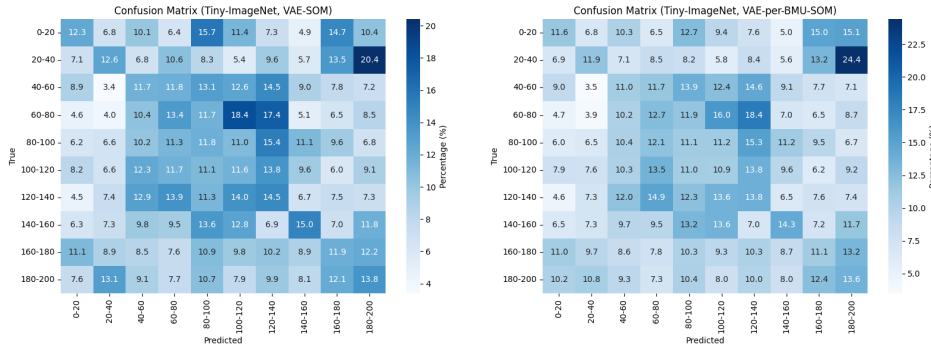


Figure 7: Confusion matrices for TinyImageNet with VAE (FT global) (left) and VAE (FT BMU specific) (right).

beneficial for optimizing hyperparameters or performing early stopping. Appendix E provides a full presentation of SOM unit vector visualizations after each class (or for every 10 classes, in the case of CIFAR-100) for each dataset.

Across datasets, we observe that the SOMs initially form a well-distributed representation of the initial class. After training on the next class, part of the map is overwritten by the new class, while some regions preserve some information from previous classes, which were synthetically generated from the prior SOM iteration. As new tasks are learned, this process continues and representations for prior classes become more compressed in the map; by the time the final class is introduced, the representation is significantly shifted. However, distinct clusters of previously-learned classes remain in various regions for all datasets, showing that our method maintains earlier class representations.

Figures 9 and 10 show a visualization of the unit vectors after each task for the SOMs generated for MNIST and Fashion-MNIST, respectively. Each image in the grid represents the SOM BMU unit vector after observing class 0, 1, and up to class 9 sequentially in the one-class-at-a-time learning setup. The 784 dimension unit vectors are converted back to 28x28 tensors of the original image sizes, and then scaled back to pixel space. Similarly, Figures 11 and 12 show the SOM BMU unit vectors after being decoded back to images by the VAE. CIFAR-10 shows the SOM state after each class, and CIFAR-100 shows the SOM after every 10 classes. These images show the methodology retains regions for each learned class within the SOM.

## F SOM ARCHITECTURE AND ENCODER-DECODER INTEGRATION

For datasets trained from scratch, we employed the residual VAE in Table 11 which compresses CIFAR images into latent codes of size  $32 \times 2 \times 2$ ,  $64 \times 2 \times 2$ , or  $128 \times 2 \times 2$ , reconstructed symmetrically with ResUp blocks (12.6M parameters for the 32-dimensional VAE variant). For pretrained





Figure 8: Visualization of decoded SOM unit vectors from the VAE (FT global) (left) and CLIP-SOM (right) models after the third classes on CIFAR-10 for a 25x25 SOM.

Table 11: Architecture of the VAE model used for CIFAR-10/100 (input size:  $3 \times 32 \times 32$ ). ResDown and ResUp denote residual blocks with downsampling and upsampling, respectively. All activations are ELU except for the output layer, which uses Tanh.

Layer	Kernel / Type	Stride / Scale	Activation	Skip	Output Shape	Params
<i>Encoder</i>						
Conv2d (Input)	$3 \times 3$ / Conv	1	–	No	$32 \times 32 \times 32$	896
ResDown Block 1	$3 \times 3$ / $\times 2$	$2 + 1$	ELU	Yes	$64 \times 16 \times 16$	46.3K
ResDown Block 2	$3 \times 3$ / $\times 2$	$2 + 1$	ELU	Yes	$128 \times 8 \times 8$	184.6K
ResDown Block 3	$3 \times 3$ / $\times 2$	$2 + 1$	ELU	Yes	$256 \times 4 \times 4$	737.9K
ResDown Block 4	$3 \times 3$ / $\times 2$	$2 + 1$	ELU	Yes	$512 \times 2 \times 2$	2.36M
ResBlock	$3 \times 3$ / $\times 2$	$1 + 1$	ELU	Yes	$512 \times 2 \times 2$	2.36M
Conv2d ( $\mu$ )	$1 \times 1$ / Conv	1	–	No	$32 \times 2 \times 2$	16.4K
Conv2d ( $\log \sigma^2$ )	$1 \times 1$ / Conv	1	–	No	$32 \times 2 \times 2$	16.4K
<i>Decoder</i>						
Conv2d (Latent)	$1 \times 1$ / Conv	1	ELU	No	$512 \times 2 \times 2$	16.9K
ResBlock	$3 \times 3$ / $\times 2$	$1 + 1$	ELU	Yes	$512 \times 2 \times 2$	2.36M
ResUp Block 1	$3 \times 3$ / $\times 2$	Upsample $\times 2$	ELU	Yes	$256 \times 4 \times 4$	627.4K
ResUp Block 2	$3 \times 3$ / $\times 2$	Upsample $\times 2$	ELU	Yes	$128 \times 8 \times 8$	295.2K
ResUp Block 3	$3 \times 3$ / $\times 2$	Upsample $\times 2$	ELU	Yes	$64 \times 16 \times 16$	184.5K
ResUp Block 4	$3 \times 3$ / $\times 2$	Upsample $\times 2$	ELU	Yes	$32 \times 32 \times 32$	82.9K
Conv2d (Output)	$5 \times 5$ / Conv	1	Tanh	No	$3 \times 32 \times 32$	2.4K
<b>Total Parameters</b>						<b>12.6M</b>

settings, the encoder was replaced with the frozen CLIP ViT-B/32 visual tower, while the decoder (Table 12) upsampled the 512-D embedding through residual and attention blocks to match dataset resolution ( $32 \times 32$  for CIFAR-10/100,  $64 \times 64$  for TinyImageNet). In the fine-tune setting, only the final transformer block of CLIP was unfrozen. Latent vectors from either encoder were projected onto SOM grids between  $20 \times 20$  and  $40 \times 40$  units, with  $40 \times 40$  providing the best balance of capacity, stability and, computation. Each SOM unit stored running statistics (mean, variance, covariance), which were used to generate replay samples via the respective decoder.

1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133

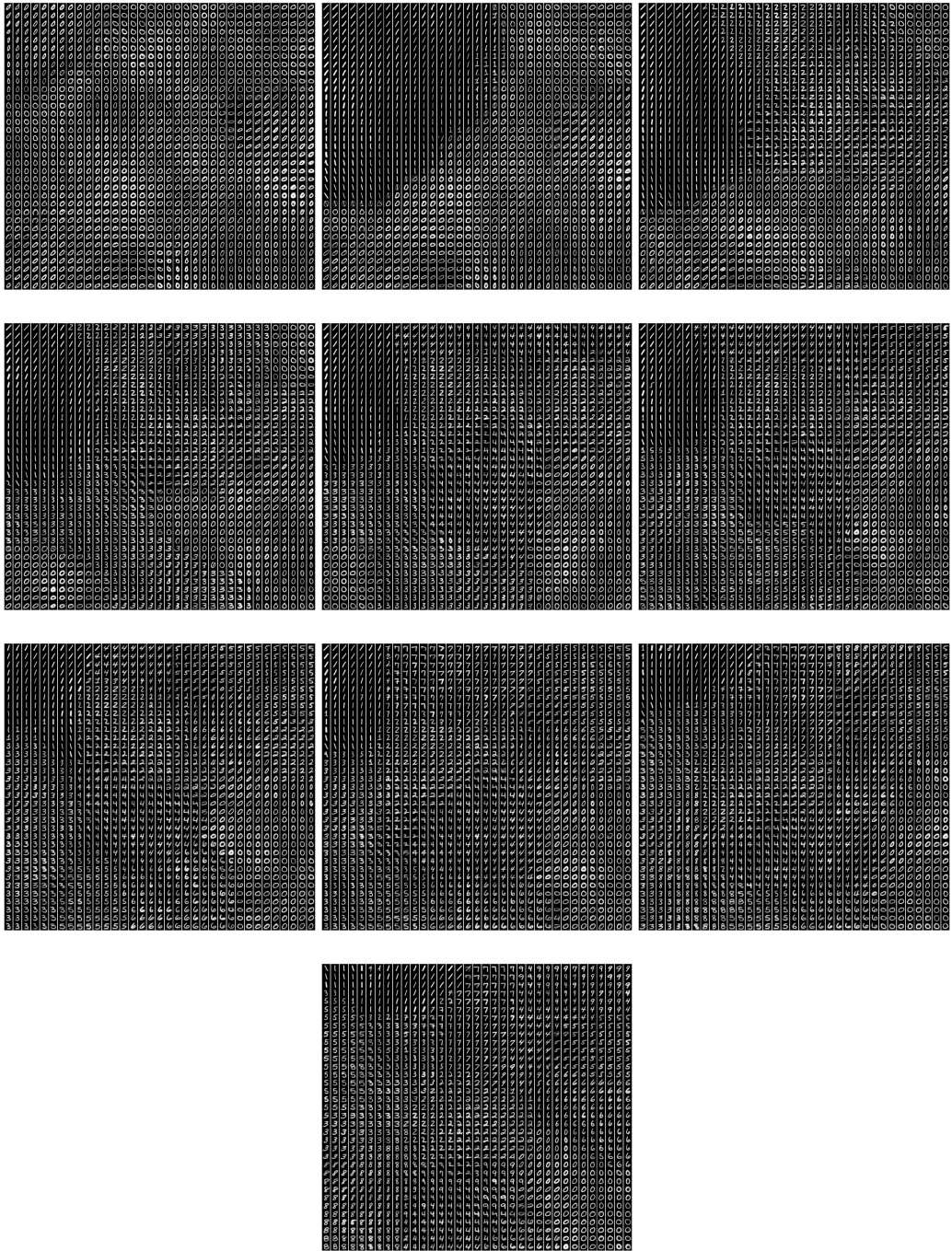


Figure 9: Visualization of the SOM unit vectors after each task for MNIST (35x35 SOM).



1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187

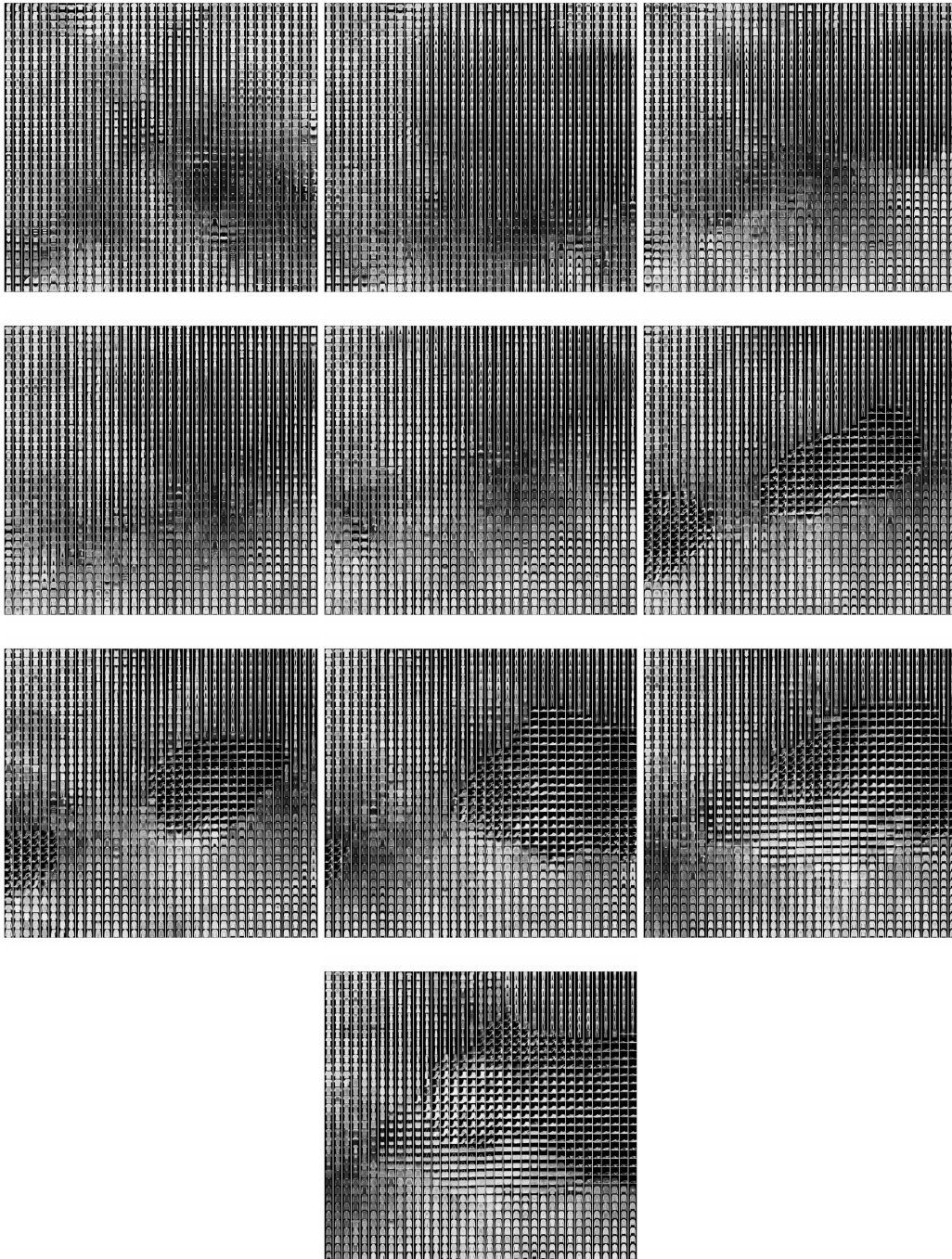


Figure 10: Visualization of SOM unit vectors after each task for Fashion-MNIST (35×35 SOM).

1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241



Figure 11: Visualization of decoded SOM unit vectors after each task on CIFAR-10 (40x40 SOM).



1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295

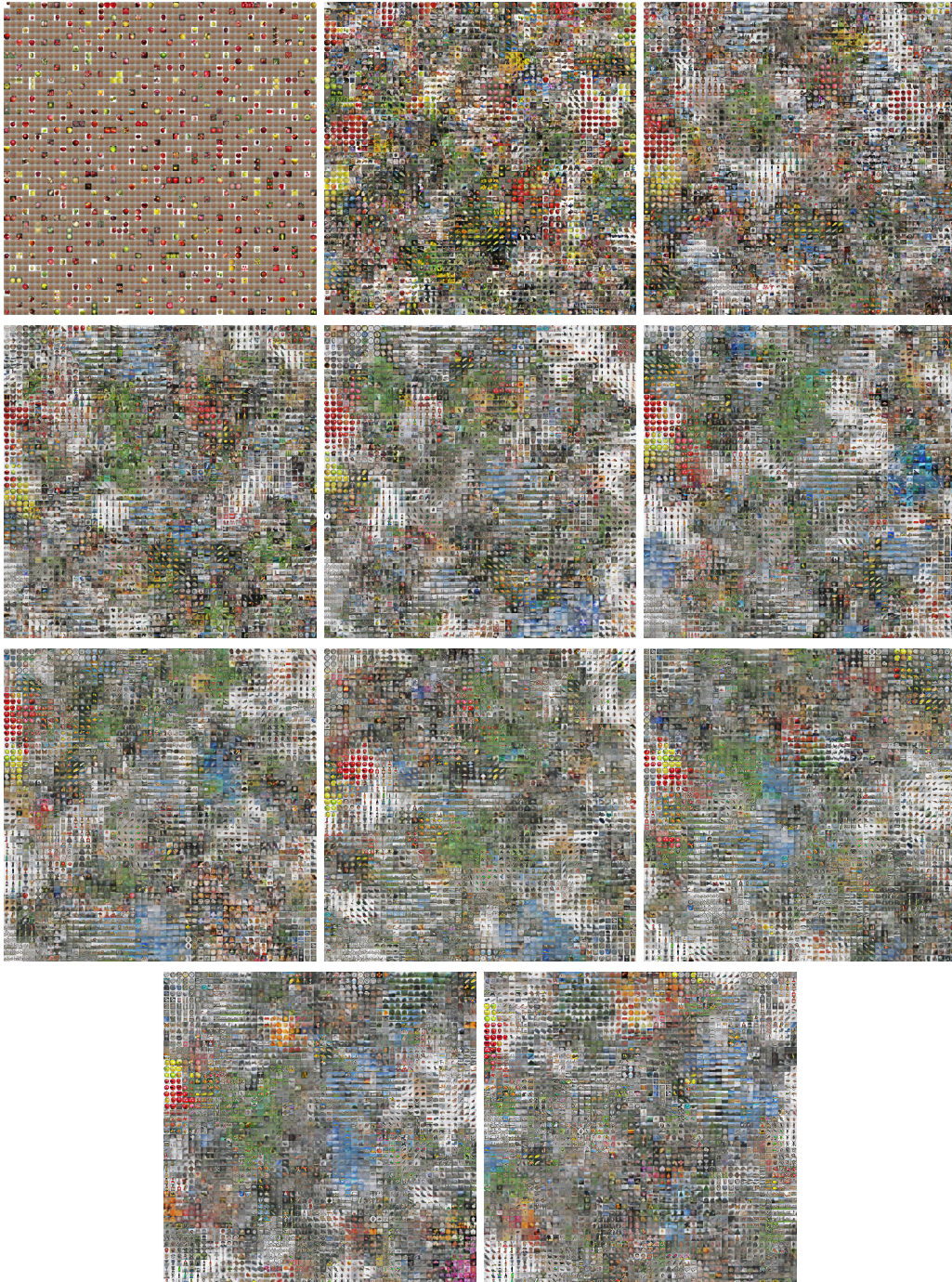


Figure 12: Visualization of decoded SOM unit vectors after every 10 tasks on CIFAR-100 (50x50 SOM).

Table 12: Decoder Architecture used for CLIP+SOM experiments. The encoder is the CLIP ViT-B/32 visual tower (pretrained) and is *frozen* by default; when `finetune_last` is enabled, only the last transformer block of ViT-B/32 is unfrozen. The decoder upsamples a single CLIP embedding to the target RGB image in  $[-1, 1]$ . GN denotes GroupNorm(32).

Layer	Operation / Details	Blocks	Activation	GN	Output Shape
Linear (embed $\rightarrow 4 \times 4$ )	FC( $D \rightarrow \text{base\_ch} \cdot 4 \cdot 4$ )	–	–	Yes	$B \times 1024 \times 4 \times 4$
ResBlock	Conv $3 \times 3$ (GN+SiLU)	1+1	SiLU	Yes	$B \times 1024 \times 4 \times 4$
Upsample	ConvTranspose2d $4 \times 4$	$\times 2$	–	Yes	$B \times 1024 \times 8 \times 8$
ResBlock (ch $\downarrow$ )	Conv $3 \times 3$ (GN+SiLU)	1+1	SiLU	Yes	$B \times 512 \times 8 \times 8$
Upsample	ConvTranspose2d $4 \times 4$	$\times 2$	–	Yes	$B \times 512 \times 16 \times 16$
ResBlock (+Attn)	Conv $3 \times 3$ + SelfAttn2d	1+1	SiLU	Yes	$B \times 512 \times 16 \times 16$
ResBlock (ch $\downarrow$ )	Conv $3 \times 3$ (GN+SiLU)	1+1	SiLU	Yes	$B \times 256 \times 16 \times 16$
Upsample	ConvTranspose2d $4 \times 4$	$\times 2$	–	Yes	$B \times 256 \times 32 \times 32$
ResBlock (+Attn)	Conv $3 \times 3$ + SelfAttn2d	1+1	SiLU	Yes	$B \times 256 \times 32 \times 32$
ResBlock (ch $\downarrow$ )	Conv $3 \times 3$ (GN+SiLU)	1+1	SiLU	Yes	$B \times 128 \times 32 \times 32$
Upsample	ConvTranspose2d $4 \times 4$	$\times 2$	–	Yes	$B \times 128 \times 64 \times 64$
ResBlock	Conv $3 \times 3$ (GN+SiLU)	1+1	SiLU	Yes	$B \times 128 \times \text{target} \times \text{target}$
Output head	GN $\rightarrow$ SiLU $\rightarrow 3 \times 3 \rightarrow$ Tanh	1	Tanh	Yes	$B \times 3 \times \text{target} \times \text{target}$

**Notes:** Encoder output dim  $D=512$  for ViT-B/32. SelfAttention2d uses 4 heads with GN(32). By default, only the decoder is trainable; enabling `finetune_last` unfreezes the last ViT block. Decoder output target size is set to match dataset resolution ( $32 \times 32$  for CIFAR-10/100,  $64 \times 64$  for TinyImageNet).