Appendix

A First training stage of AnoMem (representation learning)

Algorithm 1 AnoMem first learning stage

```
1: Input: batch size $B$, invariance transformations $\mathcal{T}$
2: Initialization: encoders $f^{(1)}, \cdots, f^{(S)}$, memory $\text{HF}^{(1)}, \cdots, \text{HF}^{(S)}$.
3: while not reach the maximum epoch do
4:     Sample image minibatch $\boldsymbol{X}$ with labels $\boldsymbol{y}$
5:     Sample augmentations $t, t'$ from $\mathcal{T}$
6:     Get augmented views $\mathbf{Z}^{(0)} \leftarrow t(\mathbf{X})$ and $\mathbf{Z}'^{(0)} \leftarrow t'(\mathbf{X})$
7:     for $s = 1 \cdots S$ do
8:         $\mathbf{Z}^{(s)} \leftarrow f^{(s)}(\mathbf{Z}^{(s-1)})$ and $\mathbf{Z}'^{(s)} \leftarrow f^{(s)}(\mathbf{Z}'^{(s-1)})$
9:         Sample $\lfloor H^{(s)} W^{(s)} r^{(s)} \rfloor$ vectors $Z_{i,j}^{(s)}$ from $\mathbf{Z}^{(s)}$
10:        Retrieve each $Z_{i,j}^{(s)}$ memory prototypes using Eq. (1) with the $s^{\text{th}}$ scale memory layer.
11:    Compute $\mathcal{L}_{\text{COM-MS}}$ from Eq. (4)
12:    Gradient descent on $\mathcal{L}_{\text{COM-MS}}$ to update $f^{(1)}, \cdots, f^{(S)}$and $\text{HF}^{(1)}, \cdots, \text{HF}^{(S)}$.
13: Output: Encoder network $f$, and the multi-scale memory prototypes from $\text{Mem}^{(1)}, \cdots, \text{Mem}^{(S)}$.
```

B Implementation details

Optimization. Training is performed under SGD optimizer with Nesterov momentum (Sutskever et al., 2013), using a batch size of $B = 1024$ and a cosine annealing learning rate scheduler (Loshchilov & Hutter, 2017) for both of the stages.

Data augmentation. For the contrastive invariance transformations, we use random crop with rescale, horizontal symmetry, brightness jittering, contrast jittering, saturation jittering with Gaussian blur and noise as in SimCLR (Chen et al., 2020). It is worth noting that we do not need specific augmentations as in CSI (Tack et al., 2020). Indeed, CSI requires *additional* shift transformations (alongside standard augmentations like SimCLR and AnoMem) to generate its pseudo OOD negatives. AnoMem performs *differently*: it directly learns prototypes of normal samples and does *not* rely on pseudo negatives or require shift augmentations.

Model design. We conducted a performance evaluation of AnoMem using different backbone architectures, including EffNet-B0, ResNet18, and ResNet50. In the context of one-vs-all evaluation settings (OC and OE) on the CIFAR dataset, we achieved consistently high performance.

EffNet-B0 emerged as the top performer, demonstrating superior results, followed closely by ResNet50 and then ResNet18. The performance gaps between these architectures were relatively small, with EffNet-B0 outperforming ResNet50 by a margin of 0.2% to 0.3%, while ResNet50 surpassed ResNet18 by a modest margin of 0.4% to 0.6%. In our main paper, we presented the performance of AnoMem using ResNet50, which was the median performer. It is worth noting that the memory modules are scalable with the chosen backbone.

C Selection and tuning of hyperparameters

The variance loss weight λ_V was evaluated on CIFAR10 using different commonly used values. As shown in Tab. 6, AnoMem is not sensitive to λ_V. Moreover, in our tests, it remains *fixed* for all other datasets.

For the confidence weights $\lambda^{(s)}$, we compared two monotonically increasing functions linear and exponential. As we can see from the results in Tab. 7, the exponential weighting yields a better anomaly detection than the linear weights. We believe that the linear relation does not sufficiently enhance object anomalies and imposes excessive constraints during representation learning, particularly when contrasting very low-scale patterns.

The margin M enforces a norm constraint on normal and anomaly representations within a radius of less than $1/M$ and more than M, respectively. A value of 1 pushes all representations indiscriminately onto the unit sphere, while a high value leads to an ill-posed distance loss. We chose $M = 2$ as a compromise, allowing half of the unit sphere to be allocated for normal samples. Indeed, we are currently working on dynamically adjusting the value of M.

Table 6: Linear evaluation on CIFAR-10 for different λ_V.

λ_V	0.1	0.05	0.025	0.01
AUROC (%)	91.90	91.91	91.87	91.86

Table 7: OC-AD evaluation on CIFAR-10 and CIFAR-100 for different $\lambda^{(s)}$ functions.

$\lambda^{(s)}$	**CIFAR-10**	**CIFAR-100**	**avg.**
linear	91.2	85.6	88.4
exp	91.5	86.1	88.8

D Qualitative effectiveness of learned prototype

We perform in Fig. 5 a t-SNE analysis on the last-scale prototypes, along with test samples from normal and anomalous classes of CIFAR-10 after the first learning stage of AnoMem. As we can see, the representations of normal and anomalous samples are well separated, confirming the effectiveness of our memory backed representation learning. Moreover, the learnt normal prototypes are well representative of the normal class, as testified by the close overlap in representation space.

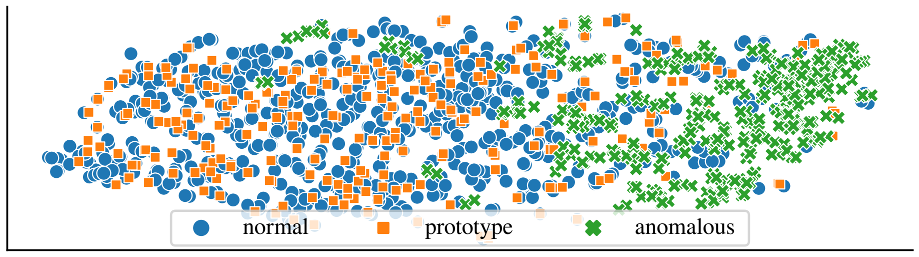

Figure 5: t-SNE of learnt prototypes, normal and anomalous samples on CIFAR10.