Appendix

A. Experiment details

A.1. Implementations

In our work, we implement the baseline methods under the setting of CLIP models, which are initially designed for conventional machine unlearning methods. Here, we present the implementation details as follows.

Notations. Given the CLIP model denoted as $g(\cdot,\cdot) = \{g^{\text{img}}(\cdot), g^{\text{txt}}(\cdot)\}$, processes image and text inputs into embeddings, e^{img} and e^{txt}. We denote the forgetting dataset as $\mathcal{D}_f$, the retaining dataset as $\mathcal{D}_r$. Each example x^{img} in both $\mathcal{D}_f$ and $\mathcal{D}_r$ has the label c, which is used as the paired text description for text encoding in CLIP. For zero-shot classification, we compute the text embedding of different classes. For each image, we compute its image embedding and the cosine similarity between the image embedding and text embedding. We take the class which has the maximum cosine similarity as the classification result.

FT. We directly fine-tune the image encoder $g^{\text{img}}(\cdot)$ on the retaining dataset. We use the Adam optimizer and set the learning rate as 10^{-6}. The batch size is set as 128. We optimize the model on the retaining set with 2 epochs.

GA. We perform the gradient ascent on the forgetting dataset, setting the learning rate as 10^{-6}. We use the Adam optimizer the optimize the model with 2 epochs.

Fisher. We compute the Fisher Information Matrix on the forgetting dataset and perturb the model parameters with noise sampled from a Gaussian distribution, where the variance is derived from the Fisher Information Matrix.

LIP. For a given image to forget, we first generate its multiple copies with injected random noise. Then, we optimize the embedding of original images and noisy images based on lipschitz constraint as follows,

$$\min \mathcal{L}_{emb} = \sum_{i=1}^{N} \frac{1}{N} \frac{\|g^{\text{img}}(x) - g^{\text{img}}(x+\epsilon_i)|}{\|\epsilon\|}, \qquad (7)$$

where we have $\epsilon \sim \mathcal{N}(0,\sigma)$.

At the same time, under the setting of zero-shot classification, we also apply the lipschitz constraint on the cosine similarity between the image embedding and text embedding as follows,

$$\min \mathcal{L}_{cls} = \sum_{i=1}^{N} \frac{1}{N} \frac{\|l(x) - l(x+\epsilon)|}{\|\epsilon\|}, \qquad (8)$$

where $l(\cdot)$ is the cosine similarity between the image embedding and the text embedding computed by possible classes.

EMMN. We jointly optimize $g^{\text{img}}(\cdot)$ on the forgetting dataset and the retaining dataset. Specifically, we maximize the loss function value on the forgetting dataset and minimize the loss function value on the retaining dataset. The learning rate is set to 1×10^{-6}. We use the Adam optimizer and fine-tune the model for 5 epochs.

Table 6. Leveraging the model merging to achieve continuous unlearning for multiple classes.

Classes	Airplane	Mobile	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck
Original	81.5	97.6	85.6	68.6	59.4	67.7	48.4	84.3	68.1	54.5
Ship (S)	78.5	82.2	85.4	59.9	41.3	62.9	47.9	73.7	0.0	78.6
Airplane (A)	0.0	87.6	78.7	72.3	48.9	51.5	46.2	78.5	60.5	64.1
Cat (C)	72.7	85.6	88.6	0.0	42.3	56.6	47.7	75.9	51.5	44.6
S+A	10.0	88.3	83.0	58.4	47.5	62.5	55.4	77.9	10.0	70.8
S+A+C	5.7	91.4	88.4	1.9	42.8	51.5	55.5	80.1	5.1	55.0

A.2. Model merging for continuous forgetting

In our method, we leverage model merging to restore the zero-shot classification capacity of the studied model while forgetting the targeted knowledge. Through our experimental exploration, we observe that two models, each forgetting one class, can be simply merged into a single model capable of forgetting both classes.We report the result in Tab. 6.

First, we use our method to obtain CLIP models that have forgotten the classes ”ship,” ”airplane,” and ”cat,” respectively. Next, we merge the models forgetting ”ship” and ”airplane” (denoted as S+A). The resulting model successfully forgets both classes. Similarly, the model denoted as S+A+C, obtained by merging models that forget ”ship,” ”airplane,” and ”cat,” successfully forgets all three classes together. An intriguing observation is that while the model performs worse on most classes, it exhibits improved performance on certain specific classes, such as *frog*. Understanding the underlying reasons for this phenomenon remains an open question for future research.