Zero-Shot Class Unlearning in CLIP with Synthetic Samples

Alexey Kravets
University of Bath,
Bath, UK
ak3095@bath.ac.uk

Vinay P. Namboodiri
University of Bath,
Bath, UK
vpn22@bath.ac.uk

Abstract

Machine unlearning is a crucial area of research. It is driven by the need to remove sensitive information from models to safeguard individuals' right to be forgotten under rigorous regulations such as GDPR. In this work, we focus on unlearning within CLIP, a dual vision-language encoder model trained on a massive dataset of image-text pairs using contrastive loss. To achieve forgetting we expand the application of Lipschitz regularization to the multimodal context of CLIP. Specifically, we smooth both visual and textual embeddings associated with the class intended to be forgotten relative to the perturbation introduced to the samples from that class. Additionally, importantly, we remove the necessity for real forgetting data by generating synthetic samples via gradient ascent maximizing the target class. Our forgetting procedure is iterative, where we track accuracy on a synthetic forget set and stop when accuracy falls below a chosen threshold. We employ a selective layers update strategy based on their average absolute gradient value to mitigate over-forgetting. We validate our approach on several standard datasets and provide thorough ablation analysis and comparisons with previous work.

1. Introduction

Motivation Machine unlearning [20] is becoming an important research area as the need to remove specific learned information from models grows. CLIP [15], a versatile model utilized in fields like robotics, content moderation, and image classification, plays a critical role in many systems. However, this widespread use also raises concerns. If CLIP has inadvertently learned to recognize sensitive or proprietary information during its training, this knowledge can spread across different applications leading to serious ethical and legal issues. For instance, if private images accidentally leaked into the model during training, causing it to recognize individuals, regulations such as the GDPR[1] mandate that such information must be erased. Additionally, in dynamic applications like object tracking or image classification, where models are regularly retrained or updated with new data, the ability to selectively unlearn outdated or incorrect information without needing to retrain the entire model from scratch could save considerable computational resources and ensure that the model remains accurate and up-to-date.

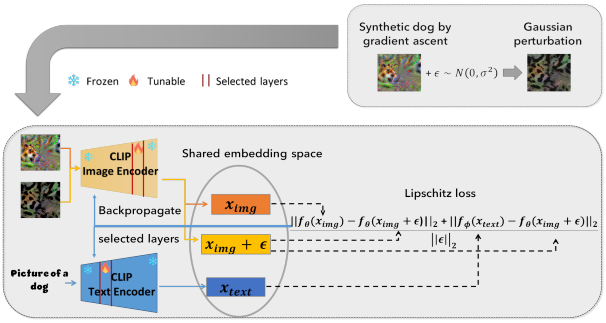

Figure 1. **Overview of the approach.** First, we generate synthetic images of a class to forget by gradient ascent. Then, we perform a Gaussian perturbation of the images and pass the original and perturbed images through CLIP image encoder and textual description of the class through CLIP textual encoder. As image and text are projected into a shared embedding space, we can use the final representation of the perturbed image as a perturbed representation of text "Picture of a dog". Finally, we apply Lipschitz regularization and backpropagate to selected layers based on their importance to visual and textual encoders.

Challenges of Unlearning CLIP: CLIP and other related vision-language models can be used for image and text-based tasks without being retrained. The unlearning for such models is especially challenging for the following three reasons: a) they have separate visual and textual encoders. Thus, even if we unlearn using the image representation, the textual modality may still be able to generalise. For instance, if we want to forget the concept of a specific breed of dog - Chihuahua and are able to use a method to unlearn in the image representation, the model

[1] https://gdpr-info.eu/art-17-gdpr/

may still be able to caption the dog Chihuahua as the information may be retained in the textual representation b) we do not have access to the data used to train the CLIP model[2]. Techniques that require availability of samples for unlearning are thus not applicable c) CLIP model has a large number of parameters hence retraining it is unfeasible. In our work, we explicitly design a method that overcomes all these challenges.

Devising a solution: In order to solve the problem, we need a basic principle that we can use for unlearning. The most straightforward method is to retrain the original model without using specific data that is intended to be forgotten. As mentioned, for a foundational model like CLIP, that is computationally expensive. The other approach could be to use random sampling of data to be retained and some samples of the data to be forgotten and using a technique like amnesiac forgetting [9] to adapt the CLIP model. As CLIP model spans many concepts, that is not practical. We next consider methods that rely only on samples of data to be forgotten. A recent method [7] shows that by retraining the model by perturbing the samples to be forgotten, one can achieve unlearning. We follow this approach and aim to extend it to address all the challenges associated with unlearning in CLIP.

The first step in using the perturbation approach is to determine the type of perturbation to apply. We compared Lipschitz with other embedding perturbation methods and found that it performs well. Consequently, we adopted this approach for unlearning. Next, we tackled the challenge of needing real data samples for forgetting. Since we lacked access to the actual data used in training the model, we addressed this problem by generating synthetic image samples using gradient ascent [16]. We further solved the problem of unlearning with the dual modalities noticing that a simple extension of [7] adding continuous Gaussian noise to discrete textual tokens to unlearn the textual encoder is not possible; thus we propose to use the image representation as a proxy for text representation to unlearn the textual encoder. We finally needed to solve the computational challenge of unlearning. We do so by selectively changing the weights of only a few layers instead of retraining the whole model. We pursued an iterative unlearning approach that allows us to precisely control the amount of unlearning required to forget the specific class while retaining the information for all other classes.

Overview of our approach: We provide an overview of our approach in Fig. 1. Let us consider that a specific class needs to be forgotten, such as the class of dog *Chihuahua*. Our approach involves generating synthetic samples of this class using gradient ascent. We then use iterative forgetting through Lipschitz regularization using the synthetic image samples and do the same for the textual representation through the joint image-text representation.

Contributions Our main contributions are summarizes as follows: a) differently from previous methods [7] our approach does not rely on any real training data to forget, thus it is truly zero-shot. b) We extend Lipschitz unlearning to CLIP in a novel way noticing that adding continuous Gaussian noise to discrete textual tokens to unlearn the textual encoder is not possible. c) We propose an iterative forgetting procedure with a clear stopping criteria based on the accuracy of the synthetic samples used for forgetting.

2. Related Work

Multimodal Unlearning Multimodal forgetting remains underexplored in the literature. Authors in [2] introduce multimodal unlearning defining it by three key properties: modality decoupling, unimodal knowledge retention, and multimodal knowledge retention. The methodology involves optimizing a multimodal model through three losses to effectively unlearn forgotten data while preserving the knowledge of retained data, satisfying these properties. However, this methodology cannot be applied to CLIP due to its non-parametric fusion of modalities. Also the method requires training data for knowledge retention.

In [22], authors attempt to induce forgetting in Stable Diffusion (SD) through attention steering. This process entails minimizing cross-attention maps from the Stable Diffusion model between latent input features and textual embeddings of concepts intended for forgetting. This disentangles textual associations from image associations of target concepts. Similarly for the SD model, authors in [8] utilize the inverse of the energy-based composition [11] to guide generation probability away from conditional towards the unconditional prediction of concepts to be forgotten by negating the predicted noise associated with the forget concept. The techniques used for generative models are not applicable to dual-encoder models such as CLIP. To the best of our knowledge, none of the existing multimodal unlearning methods specifically address forgetting in CLIP.

Machine Unlearning with Generated Data In [17], the authors generated anti-samples for classes meant to be forgotten by error maximization, which is the reverse of the minimization process employed during training. These anti-samples exhibit patterns opposite to those of the sample classes. They also train using data samples from the training data classes to be remembered. During the CLIP forgetting procedure, we also generate synthetic data. However, our approach diverges from that in [17] as our method employs loss minimization to generate synthetic data with same patterns rather than opposite patterns of the forget data. The challenge induced by the approach [17] is that a delicate balance needs to be struck between retraining with classes to be remembered vs anti-samples of classes that need to be

[2]https://github.com/openai/CLIP/issues/127

forgotten. Our approach uses regularization and synthetic samples in a more efficient manner.

Zero-shot Machine Unlearning Authors in [3] have taken the approach in [17] a step further by eliminating the dependence on real data from the classes meant to be retained. They utilize a synthesis approach similar to that in [17] for generating forget data, while the retain data are synthesized using an error minimization procedure. This solution is, therefore, zero-shot as it does not rely on any real data. Note that this method is not as practical as our approach, as for general models like CLIP, there is no explicit class that needs to be retained, but, the general capabilities needs to be retained. The work that is closest to ours is the work by Foster *et. al.* [7]. They perform forgetting via local Lipschitz regularization on unimodal vision models. They do not require the retain data but rely on real data to be forgotten for training which does not make them completely zero-shot. We extended their method to multimodal CLIP model and eliminate the need for actual data. We provide comparisons with both [7] and [3] and show that our proposed method performs better for CLIP while being more practical than either of these methods.

Non-zero-shot Machine Unlearning Methods SalUn [5] unlearns models by updating parameters based on weight saliency, needing real retain and forget data. Unrolling SGD [18] reverses gradient updates to forget specific data points, requiring an SGD-trained model, unlike CLIP's Adam training. [19] introduces a method for selectively removing features and labels from models based on the concept of influence functions but also requires real data.

3. Preliminaries

3.1. CLIP Dual Encoder Model

CLIP is a dual encoder model that consists of visual and textual components. The visual component processes images and extracts their features while the textual component processes textual descriptions and encodes them into a fixed-length vector representation. Due to the contrastive pre-training during which CLIP learns to associate images and text in a shared embedding space, CLIP is able to perform a variety of zero-shot tasks, including classification.

For classification, given N classes, they are encoded within a contextual prompt such as *"A photo of a* $\{class\}$*"* with the CLIP textual encoder f_ϕ. This results in the classifier weight matrix $W \in \mathbb{R}^{N\times d}$, where d represents the embedding dimension. When presented with a test image I_i, it is encoded using the CLIP image encoder f_θ:

$$T_i = f_\theta(I_i),\ T_i \in \mathbb{R}^d. \tag{1}$$

Following this encoding step, the dot product between the matrix W and the embedded image T_i is computed to get the zero-shot classification logits for the image I_i:

$$\text{CLIPlogits}_i = T_i W^T,\ \text{CLIPlogits}_i \in \mathbb{R}^N. \tag{2}$$

3.2. Local Lipschitz Regularization

In order to unlearn samples for a particular class, we rely on an existing technique. Our method is based on work by Foster *et. al.* [7] that utilizes the concept of Lipschitz continuity for forgetting. The idea is to locally perturb an input image that needs to be forgotten by a Gaussian noise and minimizing the ratio between the change in the outputs for the perturbed and unperturbed images and change in the input. This regularization was first proposed by Yoshida and Miyato [21] as a means for obtaining generalization. However, [7] observed that using sufficient Gaussian perturbation, the learnt response for the particular input is unlearnt. Formally, given some Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2)$ of the same dimensionality of the input image $\boldsymbol{x}$ we minimize:

$$\ell = \mathbb{E}\left(\frac{\|f_\theta(\boldsymbol{x}) - f_\theta(\boldsymbol{x}+\boldsymbol{\epsilon})\|_2}{\|\epsilon\|_2}\right). \tag{3}$$

The expectation is approximated by averaging over N perturbations. We also evaluated direct regularization on the embedding loss and observed the Lipschitz regularization to be better. We discuss it in more details in sec. 5.5.

Lipschitz regularization while being a useful way for unlearning, can be a weak signal not removing enough class information from the model. Our approach therefore relies on assuring that we ensure the unlearning of the class by validating on a set of examples synthesized for the class and using iterative unlearning for good performance.

4. Method

4.1. Setting

Given the model's training set denoted as D, in a standard machine unlearning setting we identify two subsets of D: data to retain D_r and data to forget D_f. However, our setting differs because the training data for CLIP have not been made publicly available. Consequently, we are unable to determine whether a specific data sample was used for training. Therefore, we cannot verify the forgetting of a particular sample or compare the forgetting performance to a retrained model on D_r excluding D_f. Even if we had access to the training data, achieving the latter would be infeasible due to the substantial computational resources required to retrain large-scale models like CLIP.

4.2. Extending Lipschitz Regularization to CLIP

Authors in [7] utilize local Lipschitz regularization exclusively on vision models. Our experiments reveal that updating solely the vision branch is insufficient for CLIP to

forget a selected class and adjustments to both vision and text branches are necessary. Adapting Lipschitz regularization to a dual encoder CLIP model poses a challenge as there is no direct method to perturb discrete language tokens with Gaussian noise. One potential approach involves directly modifying tokens, however, determining the degree of noise introduced to the input becomes ambiguous. Since the final layer embedding from both the CLIP vision and language branches are mapped to a shared image-text space, we can avoid perturbing the text directly. Instead, we use the perturbed image as a proxy for the perturbed text and compute Lipschitz regularization for the text branch in the same manner as for the image branch. Thus, we define our loss objective for both vision and text branches as follows:

$$\ell = \mathbb{E}\left(\frac{\|f_\theta(\boldsymbol{x_{img}}) - f_\theta(\boldsymbol{x_{img}} + \boldsymbol{\epsilon})\|_2}{\|\epsilon\|_2}\right. \tag{4}$$

$$\left. + \frac{\|f_\phi(\boldsymbol{x_{text}}) - f_\theta(\boldsymbol{x_{img}} + \boldsymbol{\epsilon})\|_2}{\|\epsilon\|_2}\right), \tag{5}$$

where f_θ is the image encoder and f_ϕ is the text encoder that output last layers embeddings, x_{img} is the image sample and x_{text} its corresponding class wrapped in the contextual prompt. As the CLIP objective ensures that the shared embeddings for image and text are close, requiring the text embedding to be close to the perturbed image embedding is a valid unlearning regularization. The expectation in the equation 5 is approximated with Monte Carlo using N perturbations for each sample.

4.3. Synthetic Forget Samples

We create synthetic forget samples by performing gradient ascent to maximize the target class [16]. Starting from random noise sample we perform the following update until the prediction on x is of a desired class:

$$x = x + \alpha \frac{\partial L(x, y)}{\partial x}, \tag{6}$$

where α is the learning rate, y the desired target class and L the loss function. We then use these synthetic forget samples to update the weights in CLIP using local Lipschitz regularization loss. These synthetic samples do not have the appearance of the sample class (examples in Appendix E) due to the simple approach we use for generation, however, these samples suffice for unlearning a class. Additional details about data generation are found in Appendix I.

4.4. Layers Update Based on the Average Gradient

We observed that updating all parameters of CLIP results in excessive forgetting. Therefore, we perform a selective update of layers based on their importance to the samples we aim to forget. To determine this importance, we calculate the average absolute gradient value of the layers and update a specific number of layers in both the vision and text branches during each iteration. Results of the ablation on forgetting with all parameters are shown in Tab. 3.

4.5. Stopping Criteria

To achieve gradual forgetting, we begin with a low value of σ and a small number of layers to update in CLIP. During forgetting we monitor the accuracy on the synthetic samples and stop the forgetting process when it falls below a predefined threshold. If during forgetting the accuracy of the synthetic data does not drop we increase both the amount of noise σ and the number of layers for a more aggressive forgetting. Full algorithm is shown in the Appendix H.

5. Experiments

5.1. Comparable Methods

As our approach is the first to be proposed for unlearning in CLIP, there are no direct comparable methods available. Therefore, we have adapted a number of methods to provide a fair evaluation of our approach. These are:

L2 embedding regularization loss (Emb) Similarly to the method outlined before we perturb the inputs with a Gaussian noise but this time, instead of Lipschitz regularization we utilize L2 regularization that is defined as follows:

$$M = \mathbb{E}(\|f_\theta(\boldsymbol{x_{img}}) - f_\theta(\boldsymbol{x_{img}} + \boldsymbol{\epsilon})\|_2 + \tag{7}$$

$$\|f_\phi(\boldsymbol{x_{text}}) - f_\theta(\boldsymbol{x_{img}} + \boldsymbol{\epsilon})\|_2) + \tag{8}$$

$$\alpha \cdot \|f_\theta(\boldsymbol{x_{img}}) + f_\phi(\boldsymbol{x_{text}})\|_2. \tag{9}$$

As only synthetic forget data is used it is a zero-shot method like ours (denoted *ZS* in the results Tab. 1).

Amnesiac forgetting with synthetic data (Amns) We adapt the approach from [9] to the multimodal setting fine-tuning CLIP with the contrastive loss. We replace the labels corresponding to the forget class randomly with a different label using synthetic data. To keep it zero-shot we do not use the retain data but only train with the forget data.

Error Minimization-Maximization Noise (EMMN) We adapt the method in [3] to multimodal setting learning retain and forget samples through loss minimization and maximization respectively and train the model on them. As the method does not require any real data it is zero-shot.

Unimodal Lipschitz (ULip) We perform forgetting only on the visual encoder of CLIP as in [7] using image perturbation and local Lipschitz regularization. We run the method using **real** data to forget. As it requires real data it is not completely zero-shot (denoted *semi ZS* in Tab. 1).

Amnesiac forgetting with real data including retain data (AmnsRetain) Similarly to *Amns* [9] described earlier we replace the labels corresponding to the forget class randomly with a different label using **real** data. This time we include the retain real data from the dataset to which the label to forget belongs to. As this method uses the data to retain it is not zero-shot (denoted *not ZS* in Tab. 1).

SalUn SalUn [5] utilizes the forget data to compute the weights saliency which are used to select the parameters to update enabling unlearning. We extend SalUn to CLIP using the version of SalUn for image classification with random labeling as in the paper. This method is also *not ZS*.

5.2. Datasets

We assess CLIP's forgetting using four high-quality fine-grained datasets: Caltech101 [6] consists of images belonging to 101 distinct categories containing examples of objects or scenes. StanfordCars [13] includes images of cars categorized into different makes and models. OxfordFlowers [14] comprises images of flowers from 102 species while StanfordDogs [12] contains images of dogs categorized into different breeds. These datasets comprise images spanning various categories with minimal overlap between them, thus we do not need to filter for similar classes to the forget class across different datasets during evaluation.

5.3. Implementation Details

We perform our experiments on CLIP with ResNet50 [10] as visual encoder. Experiments with ViT [4] and other implementation details can be found in Appendix F and K respectively.

5.4. Evaluation

As we mentioned previously, we are unable to compare CLIP performance on a forget class to the retrained version of the model without the forget data as we effectively do not know whether a certain sample was used to train CLIP due to the data being not open sourced. However, even if the data were open sourced the computational power required to retrain such a big model that relies on a huge amount of data for its zero-shot capabilities would be a challenge. Therefore, after the forgetting procedure we will evaluate CLIP's classification performance on the selected class for forgetting, the remaining classes from that dataset and the classification performance on the remaining three datasets. It's important to highlight that we aim for the accuracy on the forget class to be as low as possible, while maintaining similar accuracy levels on the remaining classes of dataset the forget class belongs to and all other datasets compared to before the application of the forgetting procedure. We summarize this information in one number for an easier comparison computing the average score. To calculate it we compute the normalized reduction in accuracy for the target class to be forgotten, denoted as A_{cl}, and the normalized reduction in accuracy for the other classes, denoted as A_{ds}, across the N datasets examined. Average score is then computed as follows:

$$\text{Avg. Score} = \frac{1}{N+1}((1 - A_{cl}) + \sum_{ds} A_{ds}). \quad (10)$$

This score varies between 0 in case unlearning is complete with accuracy of the target class of 0 while maintaining all not targeted classes accuracy on the same level and 1 when all targeted and not targeted classes are unlearned. We aim to obtain a **small** average score.

5.5. Results

Comparison across different methods In Tab. 1 we present the results of different forgetting methods averaging across three classes on four selected datasets. Full results can be found in Appendix A. *Method* column refers to the method used in the experiments. *Forgetting Type* refers to whether the method is zero-shot (ZS), indicating that no real data were used; semi zero-shot (semi ZS), where real data were used for forgetting; and not zero-shot (not ZS), where both retain and forget data were real. *Dataset* column specifies the dataset from which the class to be forgotten was selected, while the *Avg. Target Class acc.* denotes the average accuracy of the target class before (BF) and after (AF) forgetting. *Avg. Other Classes acc.* indicates the average accuracy on the other classes in the dataset, excluding the class to be forgotten. The eight columns after that display the results on the remaining datasets reported both before and after forgetting. Finally, *Avg. Score* is the aggregated metrics described previously. We observe that our forgetting procedure, referred to as *Lip* has proven successful as indicated by a notable decrease in accuracy for the targeted classes, often approaching zero. Conversely, the accuracy for the remaining classes and other datasets remain largely unaffected after applying the forgetting procedure. This, indeed, results in the lowest average score

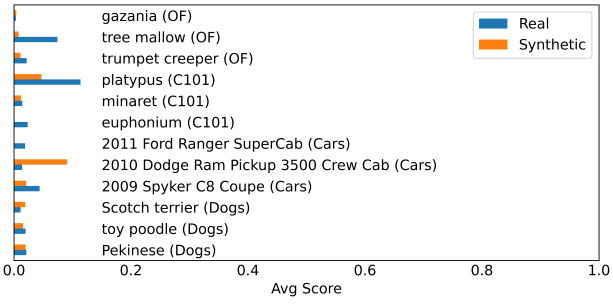

Figure 2. Comparing the average scores of unlearning with Lip method using Synthetic vs Real data.

Table 1. We compare our method (Lip) to six methods averaging across three classes for four selected datasets. We aim to minimize *Avg. Target Class acc. AF* while maintaining *Avg. Other Classes acc. AF* and other datasets at a similar level to that before forgetting (BF). We bold the best results comparing only among the first four methods that are zero-shot methods for a fair comparison.

Method	Dataset	Forgetting Type	Avg. Target Class acc.		Avg. Other Classes acc.		Avg. StanfordCars		Avg. StanfordDogs		Avg. Caltech101		Avg. OxfordFlowers		Avg. Score (↓)
			BF	AF	BF	AF	BF	AF	BF	AF	BF	AF	BF	AF	
Lip (Ours)	StanfordCars	ZS	0.397	0.056	0.558	0.551	-	-	0.517	0.513	0.857	0.86	0.661	0.653	**0.034**
Emb	StanfordCars	ZS	0.397	0.087	0.558	0.536	-	-	0.517	0.51	0.857	0.85	0.661	0.649	0.06
Amns [9]	StanfordCars	ZS	0.397	0.357	0.558	0.498	-	-	0.517	0.505	0.857	0.863	0.661	0.653	0.208
EMMN [3]	StanfordCars	ZS	0.397	0.0	0.558	0.054	-	-	0.517	0.043	0.857	0.424	0.661	0.069	0.644
ULip [7]	StanfordCars	semi ZS	0.397	0.127	0.558	0.457	-	-	0.517	0.502	0.857	0.848	0.661	0.639	0.115
AmnsRetain [9]	StanfordCars	not ZS	0.397	0.04	0.558	0.711	-	-	0.517	0.509	0.857	0.881	0.661	0.622	0.035
Salun [5]	StanfordCars	not ZS	0.397	0.063	0.558	0.712	-	-	0.517	0.491	0.857	0.862	0.661	0.574	0.068
Lip	StanfordDogs	ZS	0.593	0.048	0.516	0.516	0.558	0.558	-	-	0.857	0.866	0.661	0.655	**0.018**
Emb	StanfordDogs	ZS	0.593	0.261	0.516	0.479	0.558	0.554	-	-	0.857	0.836	0.661	0.621	0.121
Amns	StanfordDogs	ZS	0.593	0.327	0.516	0.465	0.558	0.556	-	-	0.857	0.848	0.661	0.643	0.138
EMMN	StanfordDogs	ZS	0.593	0.0	0.516	0.053	0.558	0.107	-	-	0.857	0.493	0.661	0.107	0.594
ULip	StanfordDogs	semi ZS	0.593	0.429	0.516	0.47	0.558	0.539	-	-	0.857	0.842	0.661	0.641	0.179
AmnsRetain	StanfordDogs	not ZS	0.593	0.044	0.516	0.663	0.558	0.521	-	-	0.857	0.838	0.661	0.61	0.048
Salun	StanfordDogs	not ZS	0.593	0.043	0.516	0.661	0.558	0.502	-	-	0.857	0.835	0.661	0.602	0.057
Lip	Caltech101	ZS	0.839	0.081	0.857	0.865	0.558	0.557	0.517	0.52	-	-	0.661	0.657	**0.021**
Emb	Caltech101	ZS	0.839	0.131	0.857	0.83	0.558	0.546	0.517	0.501	-	-	0.661	0.618	0.061
Amns	Caltech101	ZS	0.838	0.33	0.857	0.834	0.558	0.553	0.517	0.502	-	-	0.661	0.627	0.102
EMMN	Caltech101	ZS	0.839	0.0	0.857	0.397	0.558	0.097	0.517	0.081	-	-	0.661	0.13	0.602
ULip	Caltech101	semi ZS	0.839	0.666	0.857	0.854	0.558	0.56	0.517	0.509	-	-	0.661	0.652	0.165
AmnsRetain	Caltech101	not ZS	0.839	0.0	0.857	0.925	0.558	0.526	0.517	0.505	-	-	0.661	0.636	0.023
Salun	Caltech101	not ZS	0.839	0.0	0.857	0.924	0.558	0.528	0.517	0.502	-	-	0.661	0.642	0.022
Lip	OxfordFlowers	ZS	0.848	0.0	0.659	0.645	0.558	0.557	0.517	0.51	0.857	0.868	-	-	**0.008**
Emb	OxfordFlowers	ZS	0.848	0.442	0.659	0.625	0.558	0.553	0.517	0.505	0.857	0.85	-	-	0.122
Amns	OxfordFlowers	ZS	0.848	0.388	0.659	0.592	0.558	0.54	0.517	0.487	0.857	0.835	-	-	0.135
EMMN	OxfordFlowers	ZS	0.848	0.0	0.659	0.121	0.558	0.121	0.517	0.112	0.857	0.676	-	-	0.519
ULip	OxfordFlowers	semi ZS	0.848	0.691	0.659	0.59	0.558	0.549	0.517	0.488	0.857	0.845	-	-	0.201
AmnsRetain	OxfordFlowers	not ZS	0.848	0.059	0.659	0.922	0.558	0.553	0.517	0.51	0.857	0.866	-	-	0.018
Salun	OxfordFlowers	not ZS	0.848	0.059	0.659	0.924	0.558	0.534	0.517	0.503	0.857	0.857	-	-	0.028

across all examined methods. In comparison, the L2 embedding loss referred to as *Emb* appears to be more aggressive than *Lip*, not only erasing target knowledge but also impacting knowledge about other classes. We also tested L1 regularization loss that showed even more aggressive behaviour towards not targeted classes. We believe that the reason is that by enforcing a Lipschitz constraint we ensure that the output embeddings do not change too much in response to a small perturbation in the input (**ratio Δ output / Δ input**). This is particularly useful in unlearning as it prevents the model from drastically modifying the embeddings for non-targeted classes, hence preserving knowledge about non-target classes while only modifying the targeted ones. On the contrary, L1 and L2 regularization methods focus on penalizing the **magnitude** of the embeddings - L1 for sparsity, L2 for small embeddings. Thus, they do not inherently control how the output embeddings change to an input change. Thus applying L1 or L2 might lead to less stable unlearning, affecting non-targeted classes as we saw empirically.

On the other hand *Amns* method has less forgetting power often resulting in not enough drop in accuracy of the target class and at the same time when forgetting was relatively successful results in over-forgetting on not targeted classes. The *AmnsRetain* approach, which utilizes real data for both retention and forgetting, although not directly comparable to our method being not zero-shot, enables CLIP to forget the target class. We also observe that the accuracy on *Avg. Other Classes acc. AF* is often much higher than *BF* because of the classes to retain used for fine-tuning the model to regain the knowledge lost during forgetting. However, we note that datasets to which the forget class does not belong, and whose data was not used for retention, perform less effectively compared to our method. This demonstrates how large models like CLIP where we do not have information about the training data and classes suffer from drop in the accuracy on classes not included in the retain data. Therefore, our method not only competes in forgetting without using any real data and any retention data but also surpasses *AmnsRetain* in terms of maintaining accuracy on other datasets. Similar conclusions to *AmnsRetain* can be drawn for *SalUn*. The *EMMN* method, while often facilitating forgetting of the target class experiences significant decrease in accuracy, both on the not targeted classes of the dataset from which the forget class was picked and on other datasets. Finally, *ULip* is not only not powerful enough to forget the target class but it also destroys knowledge not related to the target class resulting in a substantial drop on other classes of both related and unrelated datasets to the forget class. We attribute this phenomenon to asymmetric forgetting where attempting to erase knowledge in only one encoder disrupts the connection between the two encoders

Table 2. Retrieval results showing precision@k for k of 1, 5, 10.

Retrieval Type	Model	Precision@1 (↓)	Precision@5 (↓)	Precision@10 (↓)
IfT	CLIP original Avg.	0.833	0.683	0.583
IfT	CLIP forget Avg.	**0.08**	**0.23**	**0.191**
IfI	CLIP original Avg.	0.417	**0.367**	**0.317**
IfI	CLIP forget Avg.	**0.333**	**0.367**	0.325

and consequently affects the projection in the shared embedding space.

Comparison of our method with real and synthetic data We compare our method when using real and synthetic data on three classes for four different datasets in Fig. 2 with full details in Appendix C. We see that forgetting yields similar average scores for synthetic and real data.

Multiple classes unlearning For experiments on unlearning multiple classes please refer to the Appendix D.

Identity unlearning In line with our motivation of the right to be forgotten, we assess if our unlearning method enables identity unlearning using PinsFaces [1] dataset that contains 105 celebrity faces. These results are presented in the Appendix G.

5.6. Verification of Forgetting Success

The accuracy achieved on synthetic forget data should serve as a measure of how effectively the model has forgotten a class. We find that for this indicator to be consistent the probability of the predicted class on synthetic samples need to be close to the probability of the real samples, otherwise there might be some discrepancy. We discuss this further in Appendix I.

5.7. Predictions Before and After Forgetting on the Target Class

In the Fig. 3 we present examples of the model's predictions before (BF) and after forgetting (AF). It's evident that post-forgetting, the model still predicts classes that closely resemble the correct ones, indicating that its general understanding of similar classes remains intact. This suggests that our method effectively targets specific knowledge of the model to remove detailed knowledge about the target class while preserving broader knowledge.

5.8. Additional Tasks

We evaluate our method for the retrieval task in addition to classification. The retrieval task involves text retrieval from the image input, image retrieval from the text input, and image retrieval from the image input. As classification can be viewed as text retrieval from an image, we present

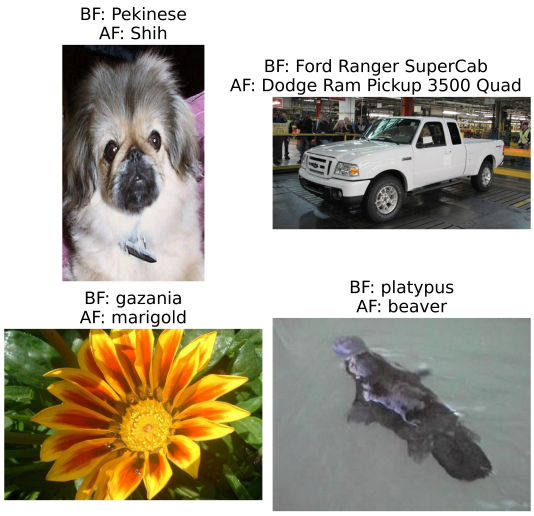

Figure 3. Predictions of the model before (BF) and after forgetting (AF) with the prediction BF representing the target class to forget.

aggregated results for the other two retrieval tasks in Tab. 2, with full results provided in Appendix J.

Image Retrieval from Text Input (IfT) We create a database from 4 datasets and perform image retrieval task given a text input. We evaluate on precision@k metric measuring the proportion of retrieved items that are relevant among top K retrieved items. This indicates the accuracy of the retrieved results. We perform our experiments with k of 1, 5 and 10. Note that the lower the precision@k the better. We see in Tab. 2 that the model is most of the times unable to retrieve images from input text.

Image Retrieval from Image Input (IfI) Similarly to above, but now we test image-image retrieval. We observe in Tab. 2 that image representation for the forget objects is mainly untouched and the model is able to find also forget classes. These results indicate that forgetting is achieved breaking the multimodal link but unimodal information still remains in the model. This was surprising and we ask whether our forgetting is successful given these results? Therefore, we checked whether the original CLIP is able to retrieve images from image input for classes the model is unable to classify, i.e. has classification accuracy of 0. In Appendix J we show that the model can identify similar features and shapes of objects without knowing the textual class. Thus, we conclude that for class forgetting, breaking the text-image association is sufficient.

6. Ablations

In this section we present ablations to understand the sensitivity of our method to changes in a) prompts to generate synthetic samples, b) evaluation templates and c) un-

Table 3. Different ablations. *Original*: results with our method (Lip) from Tab.1. *NoTextLoss*: ULip forgetting on synthetic data. *CuPLGen*: synthetic samples generated with CuPL templates. *EvalTemplChange*: evaluation with different templates.

Ablation Type	Method	Avg. Target Class acc.		Avg. Other Classes acc.		Avg. StanfordCars		Avg. StanfordDogs		Avg. Caltech101		Avg. OxfordFlowers		Avg. Score (↓)
		BF	AF	BF	AF	BF	AF	BF	AF	BF	AF	BF	AF	
Original	Lip	0.669	0.046	0.648	0.644	0.558	0.557	0.517	0.514	0.857	0.865	0.661	0.655	**0.018**
NoTextLoss	ULip	0.669	0.606	0.648	0.641	0.558	0.557	0.517	0.511	0.857	0.854	0.661	0.651	0.189
CuPLGen	Lip	0.669	0.038	0.648	0.609	0.558	0.535	0.517	0.494	0.857	0.838	0.661	0.625	0.055
EvalTemplChange	Lip	0.522	0.133	0.544	0.538	0.492	0.494	0.412	0.414	0.81	0.801	0.518	0.519	0.068

learning settings. For extra ablations and granular results please refer to Appendix B.

6.1. Textual Loss Ablation

In Tab. 3 we compare ULip (*NoTextLoss* ablation type) method and our Lip method (*Original* ablation type) using synthetic data. The only difference between these methods is the inclusion of an additional textual loss in the Lip method. The results demonstrate the critical importance of incorporating both visual and textual losses for effective forgetting in CLIP, as ULip forgetting with synthetic data proves to be highly ineffective.

6.2. Synthetic Images Generation with CuPL Template

We test how the text templates of the generated samples affect performance. We generate synthetic samples using templates from CuPL[3], e.g. for the "Pekinese" class one example is *"The image is of a small, brown and white Pekinese dog with long, flowing fur."*. We generate 64 synthetic samples and for each generated sample a random description of the class from CuPL is used. CuPL descriptions involve not only the class itself but also features of the class containing thus more information additionally to the class name. The results are shown in Tab. 3 in *CuPLGen* ablation type and more granular results in Appendix B. We see that by changing the template of the synthetic samples generation the forgetting is still successful in breaking the image-text association for the class. However, because of additional features in the template that might be shared among other classes such as *"flowing fur"* the remaining accuracy slightly decreases. Therefore, forgetting can be sensitive to the template used to generate the synthetic samples. Also note that the evaluation is still performed using the standard template.

6.3. Variation of Templates for Evaluation

In the following experiments we test the sensitivity of the model after forgetting to the change in the evaluation template. We use the synthetic samples generated with a standard template but evaluate using three different templates: *"We can see a {class} in this image"*, *"This is a representation of {class}"*, *"There is evidence of a {class} in the picture"*. This shows how sensitive the model's evaluation is to the change in template after forgetting. Note that because the evaluation template changed, so did the zero-shot classification accuracy before forgetting on CLIP. The results are shown in Tab. 3 in *EvalTemplChange* ablation type where we observe that even after changing the evaluation template forgetting is still valid across the classes we have forgotten. More granular results can be seen in Appendix B.

[3]https://github.com/sarahpratt/CuPL/tree/main/all_prompts

7. Limitations

One limitation is that it is hard to assess how well the model will perform on other classes after unlearning. Looking at *2012 Chevrolet Avalanche Crew Cab* class in Tab. 15 in the Appendix, even if forgetting is quite successful, 6% of accuracy is lost on other classes of *StanfordCars*. Note that knowledge about classes not related to cars remained fairly close to that before forgetting. Our iterative procedure can help control this trade-off between unlearning and retaining knowledge. Another limitation is that forgetting certain classes is harder and additional tuning of forgetting aggressiveness parameters and the synthetic data generation threshold might be required.

8. Conclusions

In this work we have successfully achieved class forgetting without losing knowledge on other classes in the multimodal setting of CLIP. Our experiments were conducted on four standard datasets, demonstrating that forgetting can be achieved based solely on the textual class names by generating synthetic samples of the class, without dependence on real data, thus achieving true zero-shot forgetting. Our forgetting process is iterative where we increase the number of layers to update and the strength of perturbations based on the reduction in accuracy of synthetic training data.

Acknowledgements We'd like to gratefully acknowledge Microsoft's compute support through Microsoft's Accelerating Foundation Models Research grant and the support from University of Bath for the studentship.

References

[1] Burak. Pins face recognition dataset. 7, 22

[2] Jiali Cheng and Hadi Amiri. Multimodal machine unlearning, 11 2023. 2

[3] Vikram S. Chundawat, Ayush K. Tarun, Murari Mandal, and Mohan Kankanhalli. Zero-shot machine unlearning. *IEEE Trans. Info. Forensics and Security*, 18:2345–2354, 2023. 3, 4, 6

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CVPR*, 10 2020. 5

[5] Chongyu Fan, Jiancheng Liu, Yihua Zhang, Dennis Wei, Eric Wong, and Sijia Liu. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *ICLR 2024*, 2023. 3, 5, 6

[6] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106:59–70, 04 2007. 5

[7] Jack Foster, Kyle Fogarty, Stefan Schoepf, Cengiz Öztireli, and Alexandra Brintrup. Zero-shot machine unlearning at scale via lipschitz regularization. 02 2024. 2, 3, 4, 6, 12

[8] Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models. ICCV, 06 2023. 2

[9] Laura Graves, Vineel Nagisetty, and Vijay Ganesh. Amnesiac machine learning. AAAI, 10 2020. 2, 4, 5, 6

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 12 2015. 5

[11] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *NeurIPS*, 07 2022. 2

[12] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011. 5

[13] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the 2013 IEEE International Conference on Computer Vision Workshops*, ICCVW '13, page 554–561, 2013. 5

[14] M. Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. *2008 Sixth Indian Conference on Computer Vision, Graphics Image Processing*, 2008. 5

[15] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 8748–8763. PMLR, 18–24 Jul 2021. 1

[16] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. ICLR, 2013. 2, 4

[17] Ayush K. Tarun, Vikram S. Chundawat, Murari Mandal, and Mohan Kankanhalli. Fast yet effective machine unlearning. *IEEE Trans. Neural Net. and Learn. Systems*, 07 2022. 2, 3

[18] A. Thudi, G. Deza, V. Chandrasekaran, and N. Papernot. Unrolling sgd: Understanding factors influencing machine unlearning. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroSamp;P)*, pages 303–319, Los Alamitos, CA, USA, jun 2022. IEEE Computer Society. 3

[19] Alexander Warnecke, Lukas Pirch, Christian Wressnegger, and Konrad Rieck. Machine unlearning of features and labels. In *Proc. of the 30th Network and Distributed System Security (NDSS)*, 2023. 3

[20] Heng Xu, Tianqing Zhu, Lefeng Zhang, Wanlei Zhou, and Philip S. Yu. Machine unlearning: A survey. *ACM Comput. Surv.*, 56(1), aug 2023. 1

[21] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *ICLR*, 05 2017. 3

[22] Gong Zhang, Kai Wang, Xingqian Xu, Zhangyang Wang, and Humphrey Shi. Forget-me-not: Learning to forget in text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1755–1764, June 2024. 2