SALMUBench: A Benchmark for Sensitive Association-Level Multimodal Unlearning

Supplementary Material

This document provides additional details and extended results that support the findings presented in the main paper. It includes detailed information, additional experiments, and examples of the dataset generation pipeline, model training, statistical testing methodology, and implementation specifics of all unlearning baselines. It also contains complete evaluation results, including all performance metrics reported under varying computational budgets.

7. Dataset Generation Details

This section provides additional details about the SALMU dataset construction pipeline to enhance reproducibility. The main paper provides a consolidated step-by-step summary; here we give full procedural details.

7.1. Detailed Pipeline Description

Stage 1: Anchor Seeding. We selected 1,000 reference images from the SFHQ Part 4 dataset [1], chosen for its diversity and high fidelity. These images carry no associated metadata and serve solely as identity anchors for the generation process.

Stage 2: Identity-Preserving Image Generation. For each persona, we generated approximately 100 images using IP-Adapter-FaceID Plus [29] conditioned on the anchor image. Each prompt was constructed by randomly combining phrases from multiple semantic categories, including camera angle (e.g., 'low-angle'), shot scale (e.g., 'full-body photo'), subject action (e.g., 'walking purposefully'), facial expression, and environmental setting. This automated process yielded highly varied and complex textual prompts (examples in Section 7.3), ensuring that models learn a robust, context-invariant representation of each identity.

The base image generation model produces outputs at a native resolution of 1024×1024 pixels. To introduce realistic aspect ratios and avoid domain mismatch with web-scale data, we implemented a subsequent outpainting stage. Each square image was expanded via a process guided by a ControlNet [31] model to coherently fill the new canvas area. The target aspect ratio for each image was sampled from the distribution of the DataComp corpus [8]. Finally, all outpainted images were resized to have a maximum size of 512 pixels, ensuring pre-processing consistency with the large-scale web data used for model pretraining.

Stage 3: CLIP-Based Filtering and Curation. Raw generated images underwent classifier-driven curation. We used a pretrained CLIP model [6, 12] to assign each image four zero-shot labels: (1) visual distortions, (2) multiple individuals, (3) gender, and (4) race category based on FairFace [14]. Our filtering protocol discarded any image flagged for distortion or multiple subjects. For each of the 1,000 personas, we aggregated demographic (gender, race) predictions from its remaining images, using a majority vote to establish a definitive profile, handling generation inconsistencies. We then enforced intra-identity consistency by discarding any image not conforming to its persona's established profile. Personas with fewer than 50 high-quality, consistent images were removed, filtering our pool to 774 visually and demographically coherent identities.

Stage 4: PII Assignment. For each curated persona, we generated a rich, consistent, and unique profile of sensitive attributes to simulate a comprehensive range of Personally Identifiable Information (PII), including names, jobs, locations, contact details, and financial identifiers. The process was grounded in demographic realism. Each persona's assigned ethnicity determined a plausible set of countries, from which one was sampled based on population data [24]. Subsequently, we generated a full name and city of residence using weighted samples from the Name Dataset [22] and the SimpleMaps World Cities database [23]. Each name's uniqueness was guaranteed across the dataset and checked against the Pantheon dataset [30] to avoid overlap with public figures.

Other attributes were procedurally generated for realism: blood types were sampled from real-world distributions [25], jobs were assigned with the `faker` library [7], and financial/contact information was created with structural validity. Phone numbers and IBANs followed country-specific formats, emails were derived from names, and all identifiers were unique across personas.

Stage 5: LLM Caption Paraphrasing. For each of the $\sim$60,000 curated images, we first randomly selected a sensitive attribute from its persona's profile (e.g., 'job') and generated a base sentence from a predefined template (e.g., "`{name}`'s job is `{value}`"). Then, using the LLM `gemma3:12b` [26], this sentence was paraphrased following one of five randomly selected linguistic directives: lexical substitution, word-order change, grammatical restructuring, tone shift, or creative reformulation. The LLM was constrained to include the exact name and attribute value, ensuring factual integrity. To maximize diversity, it was

also instructed to avoid repeating captions for the same fact. This process yields a rich variety of captions (e.g., "{name} is employed as a doctor" vs. "The profession of {name} is doctor"), making the underlying semantic link the target for unlearning.

7.2. Models Used in Data Generation

The image generation and outpainting pipeline leveraged several models available on the Hugging Face Hub. To maintain persona consistency, the initial images were generated by applying the `h94/IP-Adapter-FaceID`[1] adapter over a `SG161222/RealVisXL_V5.0`[2] base text-to-image model. For the subsequent outpainting stage, which introduced realistic aspect ratios, we used a process guided by the `xinsir/controlnet-union-sdxl-1.0`[3] ControlNet [31] conditioning an efficient `SG161222/RealVisXL_V5.0_Lightning`[4] base model. Finally, the generic, non-sensitive captions for the `retain_synth` set were created using the `Salesforce/blip-image-captioning-large`[5] model.

7.3. Image Generation Prompt Examples

As mentioned in the main paper, text prompts for the identity-preserving image generation were created programmatically to ensure diversity. Below are some examples of the complex prompts used. Each prompt combines elements like camera angle, shot scale, action, expression, and setting.

- *"Photorealistic low-angle wide shot photo of a single person on the top edge with negative space, in a cozy bedroom, lying, looking away from the camera, with straight and level head, pensive and thoughtful expression, sunny day, context-appropriate clothing, natural lighting, (skin texture:1.5)."*
- *"Photorealistic high-angle portrait photo of a single person on the bottom edge with negative space, in a stadium, standing with arms crossed, looking into the distance, tilting their head to the left, surprised and amazed expression, foggy day, context-appropriate clothing, natural lighting, (skin texture:1.5)."*
- *"Photorealistic side profile wide shot photo of a single person on the right edge with negative space, inside a modern office, walking purposefully, looking down, tilting their head slightly up, pensive and thoughtful expression, sunny day, context-appropriate clothing, natural lighting, (skin texture:1.5)."*

[1] https://huggingface.co/h94/IP-Adapter-FaceID
[2] https://huggingface.co/SG161222/RealVisXL_V5.0
[3] https://huggingface.co/xinsir/controlnet-union-sdxl-1.0
[4] https://huggingface.co/SG161222/RealVisXL_V5.0_Lightning
[5] https://huggingface.co/Salesforce/blip-image-captioning-large

8. Model training details

8.1. Training Data Composition.

The training data for these models relies on the splits detailed previously. For the `retain_real` component of the `retain` set, we leverage the large-scale CommonPool corpus from the DataComp benchmark [8]. This corpus contains approximately 1.3 billion image-text pairs, of which we recovered around 80%. Following the DataComp baseline strategy, we apply CLIPScore-based filtering, keeping the top 40% of pairs. This results in a high-quality dataset of approximately 400 million pairs, which we use as the foundation for both models' general knowledge. With the SALMU data splits fully defined, the training sets of both models are as follows:

- The ***Clean*** model, which represents the ideal unlearned state, is trained exclusively on the `retain` set.
- The ***Compromised*** model, which is polluted with sensitive information, is trained on the union of the `retain` and `sensitive` sets. This model serves as the starting point for all unlearning interventions.

Training Setup. Both the *Clean* and *Compromised* models adopt the ViT/B-16 architecture, consistent with the default configuration used for the large-scale flavor of the DataComp benchmark [8]. We follow the training setup from the OpenCLIP repository [12] and train both models from scratch for 32 epochs, where the *Clean* model uses `retain` (400M) and the *Compromised* model uses `retain` (400M) + `sensitive` (60K). The training process takes approximately 65 hours using 128 NVIDIA H100 GPUs. Key training hyperparameters are as follows: `batch_size`: 8192; `learning_rate`: 5e-4; and `warmup_steps`: 500.

Most of these settings match those used in the large-scale DataComp ViT/B-16 training regime, except for the number of epochs that we set to 32 as in the original CLIP model [21] and OpenCLIP [12] when trained on similar data scales (400M), ensuring both fidelity to real-world CLIP pretraining and comparability with existing CLIP model baselines.

9. Data Validation and Domain Gap Analysis

In Section 3.4 of the main paper, we discuss the validity of using synthetic faces as a proxy for real portraits. To quantify the domain gap, we compared the cosine similarity distributions of 100 real portraits from the FHIBE dataset [27] against 100 randomly sampled images from our synthetic `sensitive` set. Both sets were paired with the following generic captions to isolate visual domain differences from semantic association effects:

- *"person"*
- *"a person"*
- *"real person"*
- *"a photo of a person"*
- *"an image of a person"*

We performed a two-sample Kolmogorov-Smirnov (KS) test to compare the distributions generated by both the *Clean* and *Compromised* models. The results are as follows:

Clean Model Analysis. As visually demonstrated in Figure 6 of the main paper, the distributions for the *Clean* model overlap significantly. Quantitatively, the test yields a KS statistic of 0.0900 with a p-value of 0.8154. Since $p > 0.05$, we cannot reject the null hypothesis, indicating no statistically significant difference between the embedding distributions of real and synthetic faces.

- Real Images: $\mu = 0.2128, \sigma = 0.0175$
- Synthetic Images: $\mu = 0.2128, \sigma = 0.0141$

Compromised Model Analysis. We repeated this analysis using the *Compromised* model to ensure that the inclusion of sensitive data during training did not distort the visual domain representation. As shown in Figure 7, the distributions remain aligned. The test yielded a KS statistic of 0.1700 with a p-value of 0.1112. Again, with $p > 0.05$, the distributions are statistically indistinguishable.

- Real Images: $\mu = 0.2005, \sigma = 0.0154$
- Synthetic Images: $\mu = 0.1966, \sigma = 0.0139$

These results confirm that our synthetic dataset serves as a valid, privacy-safe proxy for real-world portraits in the context of embedding-based unlearning.

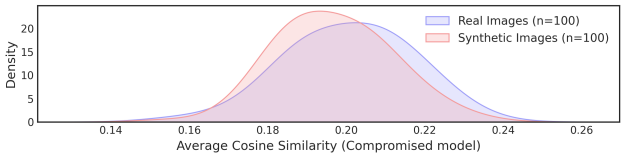

Figure 7. Average cosine similarity of Real vs. Synthetic images with generic captions for the *Compromised* model.

9.1. Clean Model Stability

To assess whether the *Clean* model's efficacy metrics reflect a stable baseline rather than a single-run artifact, we compared it against an independent OpenCLIP ViT-B/16 model trained on LAION-400M. As shown in Table 4, this independent model's sensitive-set metrics closely track our *Clean* model (e.g., RetFail 0.004 vs. 0.001), confirming that the evaluation targets represent a robust "clean regime." While strict statistical indistinguishability from a retrained model is the theoretical ideal, these results indicate that the practical goal is restoring the model to this stable baseline state.

Model	RetFail	AssocStr	ACS	IdZSC	CoreAssoc
Clean	**0.001**	0.142	0.602	0.011	0.151
LAION-400M	0.004	0.149	0.632	0.005	0.169
Compromised	0.236	0.322	0.951	0.084	0.183

Table 4. Sensitive-set efficacy metrics for the *Clean*, *Compromised*, and an independent OpenCLIP (LAION-400M) model. The similarity between *Clean* and LAION-400M confirms the target scores are stable and representative.

10. Statistical Analysis Framework

To rigorously interpret the benchmark results, we define a formal statistical framework for comparing any *Unlearned* model against the foundational *Compromised* and *Clean* models. The ultimate goal is for an *Unlearned* model to become statistically indistinguishable from its ideal target: the *Clean* model for efficacy metrics and the original *Compromised* model for utility metrics. Consequently, the ideal outcome for all statistical tests is a non-significant result (p-value ≥ 0.05), indicating that the unlearning method has perfectly reached its target state for a given metric.

Analyzing Forgetting Efficacy. The goal is for an *Unlearned* model's behavior on sensitive data to converge to that of the *Clean* model.

- For the retrieval metric (RetFail), we perform a two-sided Wilcoxon signed-rank test on the paired sample-wise ranks. We test if the distribution of ranks for the *Unlearned* model is statistically different from that of the *Clean* model. The ideal result is non-significant, as this indicates that the *Unlearned* model's retrieval performance has successfully converged to the state of the *Clean* model, which never saw the sensitive data.
- For metrics based on boolean outcomes (ACS, IdZSC), we compare the sample-wise results using McNemar's test to determine if there is a significant difference in outcome rates. The ideal result is non-significant, indicating the models are statistically equivalent.
- For metrics based on similarity scores (AssocStr, CoreAssoc), we compare the score distributions using a two-sample Kolmogorov-Smirnov (K-S) test. The ideal result is non-significant, suggesting the distributions are identical.

Analyzing Utility Impact. The goal is to preserve the general capabilities of the *Compromised* model, ensuring that the unlearning process does not introduce any significant collateral damage or unintended changes.

- To assess for any change in general knowledge (GenKnow), we use McNemar's test to compare the prediction outcomes of the *Unlearned* model against the *Com-*

promised model on ImageNet-1K. The ideal result is non-significant.

- For the remaining utility metrics (InterIdSim, IntraIdSim, VisIdInt, FragSim), we assess for any performance change using a two-sided paired t-test. This test determines if there is any significant difference (either an increase or a decrease) in the mean of the sample-wise similarity scores between the *Unlearned* and *Compromised* models. The ideal result remains non-significant, indicating that the model's performance on these utility tasks has been perfectly preserved.

11. Baselines and Methods for CLIP Unlearning

We evaluate several unlearning strategies and a finetuning baseline on SALMUBench. Since established multimodal unlearning frameworks (e.g., MultiDelete [5]) were not originally designed for CLIP's specific architecture, we performed the necessary adaptations to map their objectives to the dual-encoder setting. We evaluate each method under varying computational budgets (1×, 5×, 10×, 15×, 20×).

Generic Captions. This baseline fine-tunes the model on `forget` images paired with generic, non-sensitive captions precomputed with BLIP [16]. The training objective is to minimize the standard contrastive loss, using AdamW optimizer with a learning rate of `3e-7` and a batch size of 256.

Shuffled Captions. This method trains on batches from the `forget` set where captions are randomly shuffled, creating hard-negative pairs to break the learned associations. The model minimizes the contrastive loss on these shuffled pairs using the AdamW optimizer, a batch size of 256, and a learning rate of `5e-7`.

Direct Similarity Minimization. This method's objective is to directly minimize the mean cosine similarity between the sensitive image and text embeddings. This provides a direct "anti-similarity" signal trained with the AdamW optimizer, a batch size of 256, and a learning rate of `3e-8`.

Negative Gradient . Inspired by the approach presented in [9], it unlearns by reversing the gradient direction on the `forget` set. During training, we apply gradient ascent (maximize the loss instead of minimizing it) to push the model away from the representations learned on the `forget` data. We use batch size of 256 and AdamW optimizer with a learning rate of 3e-8.

Finetuning [9]. It finetunes the pretrained model on an 80K-example retain set (randomly sampled from the original training data, excluding the forget set), using a low learning rate of 5e-7, weight decay of 0.001, and batch size of 32.

Descent to Delete [20]. This is a weight-space unlearning method that perturbs model parameters with carefully scaled Gaussian noise to obscure the influence of specific data. The noise level is computed based on model smoothness, curvature, and the size of the original training set to ensure that the final model is statistically indistinguishable from one trained without the forgotten data.

VLUnlearn [5]. It enforces forgetting by increasing the distance between image-text embeddings for the `forget` set. The original method relies on visual-language models with cross-attention fusion layers (e.g., ALBEF or BLIP) to compute joint representations. Since CLIP lacks such fusion modules, we modify the implementation to use the average of image and text features as a proxy for the fused encoder output. We maintain the original training objectives: modality decoupling, multimodal knowledge retention, and unimodal knowledge retention. We use the default hyperparameters provided by the authors in the official repository[6].

DELETE [32]. DELETE is an unimodal class-centric unlearning method that decouples the forgetting and retention objectives. In our CLIP adaptation, the `forget` set is used to minimize the image-text similarity for sensitive pairs (forgetting), while the `retain` set is distilled from the frozen original model to preserve behavior on non-sensitive pairs (retention). We omit the original logit-masking step and instead operate directly on CLIP's cosine similarity scores. The model is trained with MSE losses on both forgetting and retention terms, using the AdamW optimizer with a learning rate of `3e-8` and batch size of 256.

CLIPErase [28]. A targeted unlearning framework for contrastive multimodal models. It jointly optimizes three objectives: (1) reducing similarity for sensitive image-text pairs (forgetting), (2) matching the original model's similarity on the retain set (retention), and (3) preserving unimodal image and text embeddings to prevent representation drift (consistency). In our implementation, we compute all three terms using MSE losses on CLIP's cosine similarity scores and embeddings, following the paper's design but without modifying the model architecture. Training uses the AdamW optimizer with a learning rate of `3e-8` and batch size of 256.

[6] https://github.com/CLU-UML/MultiDelete

Method	Budget	Forgetting Efficacy					Utility Impact				
		RetFail	AssocStr	ACS	IdZSC	CoreAssoc	GenKnow	InterIdSim	IntraIdSim	VisIdInt	FragSim
Target Scores	-	**0.001**	**0.142**	**0.602**	**0.011**	**0.151**	**0.638**	**0.321**	**0.321**	**0.305**	**0.280**
Generic Captions	1×	0.143	0.308	0.932	0.073	0.179	0.635	0.308	0.308	0.325	0.292
	5×	0.055	0.288	0.907	0.056	0.185	0.629	0.288	0.288	0.338	0.310
	10×	0.042	0.275	0.896	0.050	0.185	0.625	0.276	0.275	0.341	0.313
	15×	0.038	0.267	0.885	0.044	0.184	0.621	0.269	0.267	0.342	0.315
	20×	0.035	0.258	0.877	0.042	0.182	0.617	0.260	0.258	0.342	0.313
Shuffled Captions	1×	0.005	0.213	0.644	0.003	0.192	0.587	0.213	0.213	0.210	0.206
	5×	0.004	0.212	**0.593**	0.002	0.200	0.548	0.212	0.212	0.206	0.203
	10×	0.005	0.217	0.574	0.002	0.206	0.513	0.217	0.217	0.207	0.204
	15×	0.006	0.229	0.564	0.001	0.216	0.479	0.229	0.229	0.215	0.213
	20×	0.013	0.250	0.551	0.000	0.231	0.432	0.250	0.250	0.229	0.227
Direct Similarity Minimization	1×	0.087	0.262	0.948	0.074	0.167	**0.638**	0.264	0.263	0.283	0.259
	5×	**0.001**	-0.429	**0.593**	0.002	0.012	0.615	-0.420	-0.425	-0.038	-0.031
	10×	**0.001**	-0.860	0.529	0.000	-0.061	0.573	-0.855	-0.858	-0.263	-0.250
	15×	**0.001**	-0.944	0.520	0.000	-0.098	0.526	-0.942	-0.943	-0.323	-0.310
	20×	**0.001**	-0.967	0.516	0.000	-0.140	0.469	-0.966	-0.967	-0.355	-0.344
Negative Gradient	1×	0.179	0.313	0.929	0.072	0.189	**0.638**	**0.314**	**0.313**	**0.306**	**0.277**
	5×	0.009	0.055	0.657	0.023	0.204	0.630	0.063	0.061	0.194	0.171
	10×	0.007	-0.206	0.554	0.014	0.164	0.619	-0.201	-0.196	0.078	0.082
	15×	0.006	-0.450	0.526	**0.009**	0.143	0.604	-0.449	-0.440	-0.049	-0.024
	20×	0.006	-0.646	0.511	0.006	**0.149**	0.587	-0.649	-0.638	-0.162	-0.128
Finetuning	1×	0.004	0.218	0.647	0.006	0.222	0.646	0.220	0.220	0.287	0.266
	5×	0.003	0.208	0.646	0.005	0.212	**0.638**	0.209	0.209	0.292	0.271
	10×	0.002	0.203	0.647	0.005	0.209	0.637	0.204	0.204	0.294	0.273
	15×	0.002	0.200	0.647	0.005	0.208	0.637	0.201	0.201	0.295	0.274
	20×	0.002	**0.199**	0.648	0.004	0.207	0.637	0.201	0.201	0.295	0.274
Descent to Delete	-	0.008	0.227	0.644	0.007	0.230	0.644	0.228	0.228	0.277	0.258
VLUnlearn	5×	**0.001**	0.210	0.542	0.006	0.211	0.638	0.210	0.210	0.260	0.248
	10×	**0.001**	0.209	0.528	0.007	0.212	0.641	0.209	0.209	0.262	0.248
	15×	**0.001**	0.209	0.524	0.007	0.213	0.641	0.208	0.209	0.262	0.248
	20×	**0.001**	0.208	0.525	0.006	0.212	0.643	0.208	0.208	0.263	0.248
DELETE	5×	**0.001**	0.022	0.547	0.000	0.047	0.632	0.023	0.023	0.221	0.217
	10×	**0.001**	0.014	0.536	0.000	0.044	0.642	0.015	0.014	0.240	0.234
	15×	**0.001**	0.013	0.529	0.000	0.049	0.643	0.014	0.013	0.249	0.242
	20×	**0.001**	0.012	0.524	0.000	0.050	0.643	0.013	0.013	0.248	0.240
CLIPErase	5×	**0.001**	0.023	0.550	0.000	0.051	0.634	0.024	0.024	0.225	0.222
	10×	**0.001**	0.019	0.534	0.000	0.056	0.642	0.020	0.020	0.245	0.237
	15×	**0.001**	0.018	0.527	0.000	0.065	0.643	0.020	0.019	0.253	0.243
	20×	**0.001**	0.019	0.525	0.000	0.069	0.642	0.020	0.020	0.253	0.243

Table 5. Performance of baseline unlearning methods across different computational budgets.

12. Experimental Results

12.1. Unlearning Baselines Performance

To complement the main paper's evaluation, where we focus on the $5\times$ compute budget due to its generally favorable trade-off between forgetting efficacy and utility preservation, we provide in Table 5 the complete performance breakdown across all considered computational budgets ($1\times$ to $20\times$). This extended view offers insights into how each method scales under increasing resource availability.

In this analysis, the 'Target Scores' serve as the ideal empirical bounds: they correspond to the *Clean* model for Forgetting Efficacy metrics (representing perfect removal) and the *Compromised* model for Utility Impact metrics (repre-

		Forgetting Efficacy					Utility Impact				
Method	**Budget**	**RetFail**	**AssocStr**	**ACS**	**IdZSC**	**CoreAssoc**	**GenKnow**	**InterIdSim**	**IntraIdSim**	**VisIdInt**	**FragSim**
Clean (Reference)	**-**	**0.001**	**0.142**	**0.602**	**0.011**	**0.151**	**0.633**	**0.143**	**0.143**	**0.331**	**0.309**
Generic Captions	5×	**0.001**	**0.147**	**0.616**	**0.011**	**0.156**	0.626	**0.148**	**0.148**	**0.324**	**0.301**
Shuffled Captions	5×	0.002	0.185	0.521	0.001	0.185	0.606	0.185	0.185	0.183	0.184
Direct Sim. Min.	5×	**0.001**	-0.371	0.512	0.005	-0.259	0.611	-0.369	-0.368	0.074	0.074
Negative Gradient	5×	**0.001**	-0.114	0.529	0.007	-0.045	**0.629**	-0.116	-0.112	0.254	0.248

Table 6. Performance of 4 baseline unlearning methods applied to the *Clean* model (5× budget). The first row lists the metrics of the unmodified *Clean* model. Deviations from these values indicate unintended damage to the model's embedding space.

senting original knowledge preservation).

While some methods exhibit relatively stable behavior across budgets (e.g., Finetuning, DELETE, CLIPErase), others show strong budget-dependent dynamics. Notably, Direct Similarity Minimization performs best at low budgets but deteriorates significantly as training progresses – highlighting the method's tendency to over-optimize, resulting in excessive structural damage to the embedding space.

Similarly, methods like Generic Captions and Shuffled Captions show mild gains with larger budgets, though their forgetting performance often saturates early, suggesting limited capacity for deeper erasure without stronger supervisory signals.

Overall, we chose the $5\times$ budget for the main paper as it tends to represent the best trade-off across most methods: high forgetting performance without significant collateral damage. However, we emphasize that each method can, in principle, be re-tuned per budget setting. For example, hyperparameters like learning rate, number of epochs, or regularization strength could be adjusted to yield more optimal trade-offs at higher budgets. Nevertheless, this study intentionally fixes per-method hyperparameters across all budgets to isolate the effect of compute scale and avoid confounding factors. We leave hyperparameter tuning for budget-aware unlearning as a direction for future SALMUBench submissions.

12.2. Sanity Check: Unlearning on the Clean Model

A critical failure mode identified in our main paper is the tendency of unlearning algorithms to damage model utility regardless of whether the sensitive data was actually present. We applied unlearning baselines to the *Clean* model. Since this model represents the desired unlearned state, a safe method should effectively act as an identity function, preserving the metrics of the original *Clean* model.

Table 6 presents the results. The first row displays the original scores of the *Clean* model, serving as the reference for stability.

Analysis. The results reveal a clear dichotomy between methods, with a notable nuance regarding utility preservation:

1. **Low-Risk Methods:** Generic Captions proves to be the most stable approach regarding embedding structure, maintaining scores very close to the *Clean* model reference (e.g., InterIdSim 0.148 vs. Reference 0.143). However, it is not entirely cost-free, as it induces a slight degradation in general knowledge (GenKnow drops from 0.633 to 0.626). Shuffled Captions causes more significant drift across both utility and structural metrics, confirming that the noise from mismatched pairs is more harmful than the use of generic labels.
2. **Catastrophically Damaging Methods:** Direct Similarity Minimization and Negative Gradient fail this check significantly. Negative Gradient presents a deceptive profile: while it preserves GenKnow reasonably well (0.629), it aggressively distorts the embedding space, driving several metrics to negative values. Direct Similarity Minimization is even more destructive, causing a larger drop in utility and an even more extreme inversion of structure. The fact that these methods drive the *Clean* model's natural baseline alignment (AssocStr 0.142) into negative territory (-0.114 and -0.371) confirms they do not selectively unlearn but rather blindly disrupt the feature space.

13. Learning-Rate Sensitivity Ablation

To validate our failure mode taxonomy beyond a single hyperparameter setting, we performed a learning-rate (LR) sweep at $5\times$ budget for two representative methods: Generic Captions (classified as Type 3: Ineffective) and Negative Gradient (classified as Type 2: Over-generalized). For each method, we varied the learning rate across a wide range and measured RetFail (forgetting efficacy) and GenKnow (utility preservation).

As shown in Figure 8, the results confirm our taxonomy. Generic Captions exhibits a clear performance ceiling: increasing the learning rate degrades utility (GenKnow) without ever reaching the forgetting target (RetFail), confirming

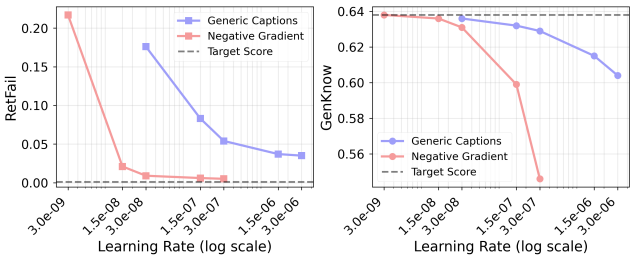

Figure 8. Learning-rate sensitivity (5× budget): RetFail (left) and GenKnow (right) for Generic Captions and Negative Gradient. Dashed lines indicate target scores. Generic Captions hits a forgetting ceiling regardless of LR, confirming Type 3 (Ineffective). Negative Gradient shows that reaching the forgetting target forces GenKnow well below the target, confirming Type 2 (Over-generalized).

its classification as Type 3 (Ineffective). Negative Gradient confirms Type 2 (Over-generalized): while intermediate learning rates might appear balanced, reaching the target RetFail forces AssocStr well below the *Clean* baseline and eventually collapses GenKnow. These results demonstrate that the identified failure modes are robust properties of the methods, not artifacts of a single hyperparameter choice.