priyam_saha17 at SVELA: A Feature-Centric Pipeline for Verifying Selective Forgetting in Large Language Models[⋆]

Priyam Saha[1,*]

[1] Associate, Cyber Risk Advisory, Grant Thornton Advisors LLC, India

Abstract

This paper presents the system developed by team priyam_saha17 for the SVELA (Selective Verification of Erasure from LLM Answers)[1] shared task at EVALITA (Evaluation of NLP and Speech Tools for Italian) 2026. SVELA addresses the problem of verifying whether a large language model has selectively forgotten specific information while retaining relevant knowledge. The problem statement is: given a large language model that has been fine-tuned and subsequently unlearned on a hidden subset of identities, determine whether a specific identity–topic pair has been retained, forgotten, or never seen during training. Rather than modifying or probing internal model parameters, the approach focuses on analyzing observable behavioral signals produced by the language models, which have been subjected to certain specific unlearning algorithms by track organisers.

On the official leaderboards, the system ranked first in all four tracks. For Task 1, mean scores of 0.3428 (1B param models) and 0.3565 (3B param models) were achieved, while for Task 2 the system obtained mean scores of 0.3345 (1B param models) and 0.3335 (3B param models). In contrast, the organizers reported baseline mean scores of 0.2813 (1B-parameter models) and 0.2855 (3B-parameter models) for Task 1, and 0.265 (1B-parameter models) and 0.2706 (3B-parameter models) for Task 2. These results indicate that feature-centric analysis of model behavior provides a robust signal for selective forgetting verification across tasks and model scales.

Keywords

Machine Unlearning, Residual Network, SVELA, EVALITA

1. Introduction

The capability of removing the influence of specific knowledge from trained machine learning models has become increasingly important in response to regulatory, ethical, and safety requirements. In the context of large language models (LLMs), this challenge is commonly addressed through machine unlearning techniques that aim to suppress or erase the influence of targeted data or concepts. However, verifying whether unlearning has been successful remains an open and underexplored problem. In practical scenarios, auditors often do not have access to the model's training data and sometimes not even its internal parameters or gradients, making behavioral verification essential.

The SVELA shared task at EVALITA 2026 explicitly focuses on this verification aspect of Machine Unlearning. Instead of asking systems to perform unlearning, SVELA evaluates whether models, that have already been finetuned and then unlearned on a hidden subset of identities by the organizers, exhibit behavioral evidence consistent with selective forgetting. Given an identity–topic pair, systems must determine whether the model has retained the information, has been unlearned on it, or has never encountered it during training.

In this work, a feature-centric verification pipeline has been proposed that reframes the selective forgetting verification as a supervised classification problem over the model's behavioral signals. By extracting structured behavioral signals from frozen LLMs and training a compact residual classifier on top of these features, the approach avoids model-specific assumptions while remaining scalable across different model sizes of 1 billion and 3 billion parameters.

[1] Official SVELA 2026 shared task website

EVALITA 2026: 9th Evaluation Campaign of Natural Language Processing and Speech Tools for Italian, Feb 26–27, Bari, Italy

[⋆] This paper describes the system submitted by team priyam_saha17 to the SVELA shared task at EVALITA 2026.

[*] Corresponding author.

impriyamsaha@gmail.com (P. Saha)
0009-0002-1167-3529 (P. Saha)

2. Related Work

Machine unlearning has been formalized as the task of removing the influence of specific data points from a trained model while preserving performance on retained data, motivated by legal and ethical requirements. Early theoretical formulations by Ginart et al. [1] introduced data deletion as a measurable property of learning algorithms. Building further on this foundation, Bourtoule et al. [2] introduced SISA (Sharded, Isolated, Sliced, Aggregated) training, a practical framework to enable efficient unlearning in deep neural networks. They formalized unlearning as producing a model that is indistinguishable from one retrained from scratch without the deleted data, and achieved this by partitioning data into independent shards and retraining only the affected shard when a deletion request occurs. These preliminary experiments unlearning showed that retraining time is significantly reduced while maintaining competitive accuracy.

In the context of large language models (LLMs), unlearning presents additional challenges due to distributed representations and large-scale pretraining. Carlini et al. [3] demonstrated that LLMs can memorize and reproduce training data, thus stressing on the need for research on controlled forgetting mechanisms.

Recent work has explored gradient-based unlearning strategies, including gradient ascent on the forget set (NegGrad) by Choi et al. [4], KL-divergence minimization to preserve retained knowledge by Maini et al. [5], and preference-based optimization methods inspired by Direct Preference Optimization introduced by Rafailov et al. [6]. In particular, Maini et al. [5] introduced Negative Preference Optimization (NPO), which reframes forgetting as aligning the model towards treating unwanted data as incorrect or low quality. Adversarial variants of these approaches have also been proposed to strengthen forgetting signals while maintaining robustness.

Despite rapid progress in unlearning algorithms, verification has received comparatively less attention. Hayes et al. [7] argue that existing evaluation protocols may provide a false sense of privacy if verification pipeline is not carefully designed and that even after unlearning, detectable traces may persist in hidden representations. It is in this context that the SVELA (Selective Verification of Erasure from LLM Answers) shared task put forward by Savelli et al. [8] explicitly defines selective forgetting verification as the problem of assessing whether a model retains or has erased specific identity-related knowledge after unlearning.

A very recent work on multilingual unlearning evaluation has been put forward in FAME (Fictional Actors for Multilingual Erasure) by Savelli et al. [9], which introduces a synthetic benchmark specifically designed for controlled analysis of Machine Unlearning in large language models. FAME comprises 1,000 fictional actor biographies and 20,000 question–answer pairs across five languages (English, French, German, Italian, and Spanish). Each biography is structured into 20 atomic facts spanning biography, career, achievements, and personal information, enabling both entity-level forgetting (removal of entire identities) and instance-level forgetting (removal of specific facts). The dataset provides two complementary splits—entity-based and topic-based—to systematically evaluate different granularity levels of forgetting. Because all identities are fictional and generated through a constrained pipeline, the benchmark guarantees that the evaluated knowledge is absent from model pretraining, allowing precise measurement of forgetting efficacy. FAME evaluates five unlearning strategies: Fine-Tuning (FT) as a baseline, Gradient Ascent (GA) to maximize loss on forgotten data, Gradient Difference (GD) to balance forgetting and retention objectives, KL Minimization (KLM) to preserve retained knowledge distributions while increasing forget loss, and Preference Optimization (PO), which aligns the model toward non-informative responses for forgotten information. Two model sizes have been leveraged for above experimentations in FAME, namely, Llama-3.2-1BInstruct and Llama-3.2-3B-Instruct. Within the broader EVALITA campaign [10], SVELA [8] builds on FAME to focus specifically on verification rather than algorithm design.

Beyond algorithmic unlearning strategies, uncertainty and confidence signals have been extensively studied as indicators of model reliability and knowledge calibration. Hendrycks et al. [11] demonstrated that softmax confidence and predictive entropy are effective for detecting misclassification and distributional shift. Practical approaches of studying LLM reliability often rely on single-pass

Table 1
Dataset statistics for SVELA Task 1 and Task 2.

Task	Retain	Forget	Test	Train Total	Test Total
Task 1	576	144	192	912	4,416
Task 2	576	144	192	912	4,416

uncertainty proxies, including predictive entropy and logit margins, which remain informative while being significantly more efficient.

Finally, the architectural choice of shallow residual networks for feature fusion is grounded in the broader success of residual learning. He et al. [12] demonstrated that residual connections improve optimization stability and gradient propagation in deep neural networks. Subsequent work has shown that residual MLP-style architectures remain competitive in structured and low-data situations, where heterogeneous feature types must be integrated effectively. When combining semantic embeddings with uncertainty-derived statistics, residual connections facilitate stable interaction between feature groups without introducing excessive model complexity.

Building on these lines of research, the present work reframes selective forgetting verification as a supervised classification problem over combined representational and uncertainty features extracted from frozen models, rather than as a modification of model parameters.

3. Task Overview

SVELA [8] consists of two tasks. Task 1 focuses on verification under controlled unlearning conditions, while Task 2 introduces more challenging settings with broader identity–topic coverage. In both tasks, participants have been provided with pretrained LLMs and labeled training data split into retain, forget, and test partitions, defined as follows:

- Retained — the identity was used to train the model and should be remembered;
- Forgotten — the identity was originally trained but has been targeted by unlearning and should be forgotten;
- Never-used — the identity was never part of training (unseen) and should not be known by the model.

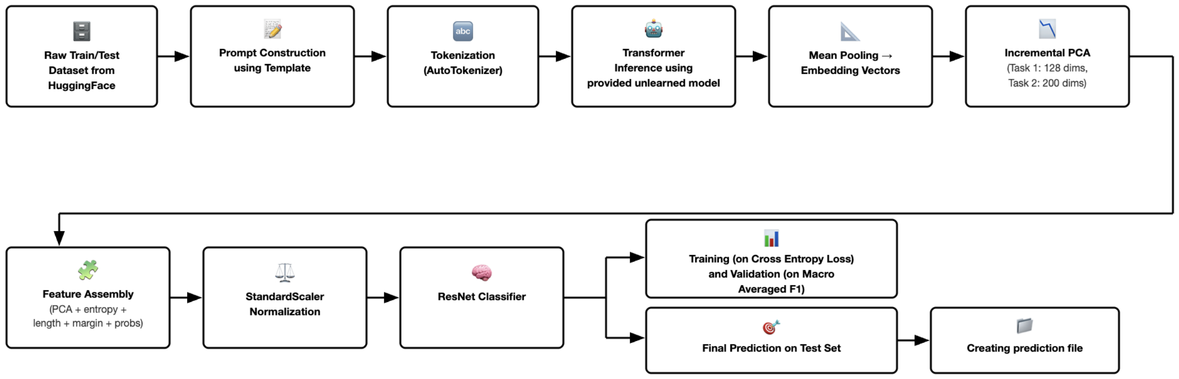
Systems must predict the correct label for unseen pairs based solely on model outputs.

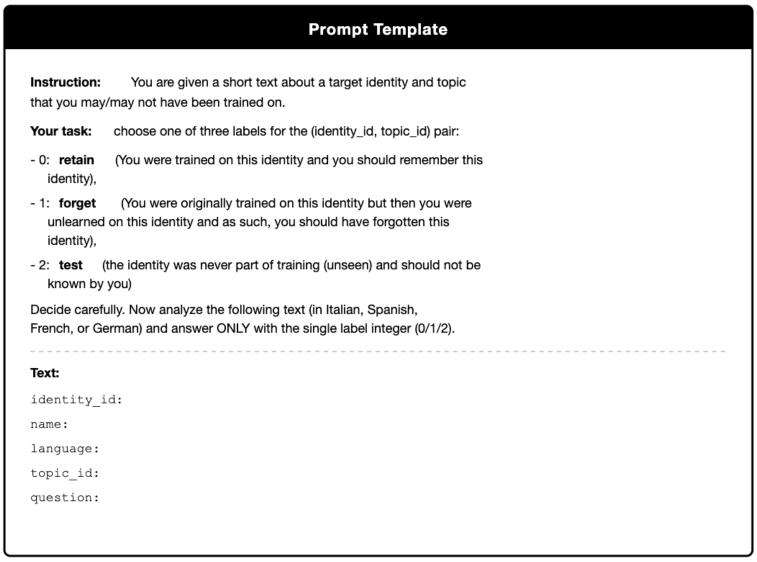
4. Methodology

The proposed pipeline consists of four stages: prompt construction, feature extraction, dimensionality reduction, and classification. A visual flowchart of the pipeline is shown in Figure 1, providing an overview of the system components and their interactions. The core design principle is to verify selective forgetting using only observable behavioral and representational signals extracted from frozen language models, without modifying their parameters. Unlearned models and labelled verification datasets had all been provided by the organizers.

4.1. Datasets and Models

Experiments are conducted using the official SVELA training and test splits released for Task 1 and Task 2. Each task contains three labels: *retain*, *forget*, and *test*. As evidenced in Table 1, the training splits are not balanced by construction. Downsampling was not applied on the majority “Retain” class, as the total number of training instances (729 after stratified splitting) is relatively small, making data reduction undesirable. Further, the official training split (912 instances per task) is further divided into


Raw Train/Test Dataset from HuggingFace
Prompt Construction using Template
Tokenization (AutoTokenizer)
Transformer Inference using provided unlearned model
Mean Pooling → Embedding Vectors
Incremental PCA (Task 1: 128 dims, Task 2: 200 dims)
Feature Assembly (PCA + entropy + length + margin + probs)
StandardScaler Normalization
ResNet Classifier
Training (on Cross Entropy Loss) and Validation (on Macro Averaged F1)
Final Prediction on Test Set
Creating prediction file



Prompt Template

Instruction: You are given a short text about a target identity and topic that you may/may not have been trained on.

Your task: choose one of three labels for the (identity_id, topic_id) pair:

- 0: **retain** (You were trained on this identity and you should remember this identity),
- 1: **forget** (You were originally trained on this identity but then you were unlearned on this identity and as such, you should have forgotten this identity),
- 2: **test** (the identity was never part of training (unseen) and should not be known by you)

Decide carefully. Now analyze the following text (in Italian, Spanish, French, or German) and answer ONLY with the single label integer (0/1/2).

Text:

```
identity_id:
name:
language:
topic_id:
question:
```

Figure 1: Overview of the proposed feature-centric verification pipeline. *Top:* End-to-end processing pipeline from raw data ingestion to prediction output. *Bottom:* Prompt template used to query the unlearned transformer model.

Table 2
Data splits used for training, validation, and testing (derived from the official SVELA splits and stratified 80/20 partition).

Task	Training	Validation	Train Total	Test
Task 1	729	183	912	4,416
Task 2	729	183	912	4,416

training and validation subsets using stratified sampling to preserve class distribution with an 80/20 ratio, resulting in 729 training and 183 validation examples per task as shown in Table 2.

Four official SVELA model checkpoints are evaluated, as shown in Table 3. The datasets utilized in this work are as follows:

- `SVELA-train-split-phase2_task1`[5]
- `SVELA-train-split-phase2_task2`[6]
- `SVELA-test-split_task1`[7]
- `SVELA-test-split_task2`[8]

Table 3
SVELA pretrained model checkpoints used in this study.

Task	Model Identifier
Task 1 (1B)	`model1b_task1_b`[1]
Task 1 (3B)	`model3b_task1_b`[2]
Task 2 (1B)	`model1b_task2_b`[3]
Task 2 (3B)	`model3b_task2_b`[4]

4.2. Prompting and Inference

All inputs are wrapped in a standardized instruction prompt in English describing the three possible labels (*retain*, *forget*, *test*). Although the question content is multilingual, the instruction layer is fixed in English to reduce variability introduced by prompt phrasing and to ensure consistent model behaviour across languages and tasks.

The underlying SVELA models are kept frozen during inference to ensure that the parameters obtained after unlearning are not further modified. The models are used strictly in evaluation mode, and hidden states as well as logits are extracted without performing gradient updates. Maximum sequence length is set to 384 tokens for Task 1 and 512 tokens for Task 2. The increased limit in Task 2 reflects the higher average input length and ensures that identity–topic information is preserved without any truncation.

4.3. Feature Extraction

Rather than relying solely on predicted labels, the proposed system extracts both representation-level and uncertainty-based signals from the frozen model.

For each example, the final-layer hidden states $H \in \mathbb{R}^{T\times d}$ are mean-pooled using the attention mask $m \in \{0, 1\}^T$:

$$\mathbf{e} = \frac{\sum_{t=1}^{T} m_t H_t}{\sum_{t=1}^{T} m_t}, \tag{1}$$

ensuring that padding tokens do not influence the representation. These embeddings capture the semantic information encoded in the model's internal state.

From the output logits, additional behavioral signals are derived. Let $\mathbf{p} = (p_1, p_2, p_3)$ denote the softmax probabilities over the three classes. Predictive entropy is computed as:

$$H(\mathbf{p}) = -\sum_{i=1}^{3} p_i \log p_i, \tag{2}$$

which measures output uncertainty. The difference between the top-1 and top-2 logits (logit margin) is used to capture confidence separation. The full softmax probability vector is retained, and the token count (derived from the attention mask) is included as a coarse measure of input length and complexity.

These features jointly capture internal semantic representation, output-level uncertainty, confidence calibration, and structural characteristics of the input.

[1] https://huggingface.co/SVELA-task/model1b_task1_b
[2] https://huggingface.co/SVELA-task/model3b_task1_b
[3] https://huggingface.co/SVELA-task/model1b_task2_b
[4] https://huggingface.co/SVELA-task/model3b_task2_b
[5] https://huggingface.co/datasets/SVELA-task/SVELA-train-split-phase2_task1
[6] https://huggingface.co/datasets/SVELA-task/SVELA-train-split-phase2_task2
[7] https://huggingface.co/datasets/SVELA-task/SVELA-test-split_task1
[8] https://huggingface.co/datasets/SVELA-task/SVELA-test-split_task2

4.4. Dimensionality Reduction

The final-layer embeddings are high-dimensional (4096 dimensions for 3B models). To reduce overfitting risk and improve computational efficiency, Incremental Principal Component Analysis (IPCA) is applied to the embedding vectors.

IPCA is chosen instead of standard PCA due to its memory-efficient partial fitting procedure, which allows processing embeddings in manageable chunks. The PCA model is fitted exclusively on the training split and subsequently applied to validation and test splits to prevent information leakage. The number of retained components is set to 128 for Task 1 and 200 for Task 2, based on explained variance analysis and empirical validation performance. Task 2 required a slightly higher number of components to preserve sufficient discriminative information.

After projection, the reduced embeddings are concatenated with entropy, token count, logit margin, and softmax probabilities to form the final feature vector. All features are standardized using a `StandardScaler` fitted only on the training data.

4.5. Dimensionality Reduction

The final-layer hidden-state embeddings were found to be high-dimensional (4096 dimensions for the 3B models), which increases the risk of overfitting given the relatively small training size (729 instances per task after stratified splitting). To reduce dimensionality while preserving the most informative variance directions, Incremental Principal Component Analysis (IPCA) is applied to the embedding vectors. Given embedding matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$, PCA computes an orthogonal projection matrix $\mathbf{W} \in \mathbb{R}^{d \times k}$ that maximizes retained variance:

$$\mathbf{W} = \arg \max_{\mathbf{W}^\top \mathbf{W} = \mathbf{I}} \mathrm{Tr}(\mathbf{W}^\top \mathbf{W}), \tag{3}$$

where denotes the empirical covariance matrix of $\mathbf{X}$. The projected embeddings are then given by $\mathbf{Z} = \mathbf{X}\mathbf{W}$.

IPCA is selected over standard PCA due to its memory-efficient partial fitting mechanism, which enables processing embeddings in manageable batches without loading the entire covariance structure into memory. The PCA model is fitted exclusively on the training split and subsequently applied to validation and test splits to prevent information leakage.

Figures 2 and 3 show the cumulative explained variance curves for Task 1 and Task 2 respectively. For Task 1, the curve exhibits a clear elbow region around 100–130 components, after which marginal variance gains diminish substantially. Retaining 128 components captures approximately 94–95% of total variance, providing a strong trade-off between compression and information preservation. In contrast, Task 2 shows a slower saturation profile, with variance distributed across a broader set of directions. Approximately 200 components are required to approach 89–90% cumulative explained variance. Empirically, retaining fewer components resulted in a measurable drop in validation F1, indicating that Task 2 embeddings contain more diffuse discriminative structure. Consequently, 200 components are retained for Task 2 to preserve sufficient representational capacity.

After projection, the reduced embeddings are concatenated with entropy, token count, logit margin, and softmax probabilities to form the final feature vector. All features are standardized using a `StandardScaler` fitted exclusively on the training data.

4.6. Residual Classifier

The final classifier is a shallow residual network designed to integrate heterogeneous feature types effectively. An input projection layer maps the feature vector to a 256-dimensional hidden space, followed by two residual blocks and a linear output layer predicting the three classes.

Each residual block follows:

$$\mathbf{h}_{l+1} = \mathbf{h}_l + f(\mathbf{h}_l), \tag{4}$$

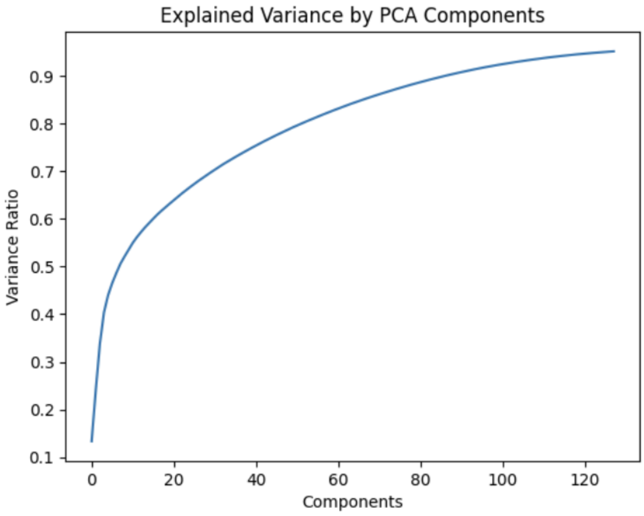


Figure 2: Cumulative explained variance for Task 1 embeddings. 128 components capture approximately 95% of total variance.

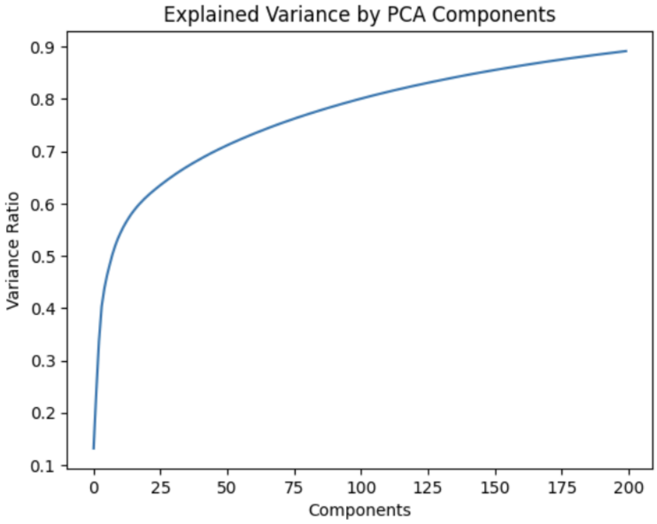


Figure 3: Cumulative explained variance for Task 2 embeddings. Variance saturates more gradually, motivating retention of 200 components.

where $f(\cdot)$ denotes a two-layer feedforward transformation with GELU activation and dropout (rate 0.1). GELU is selected instead of ReLU to provide smoother non-linearities and reduce the risk of dead neurons, that is, the “Dying ReLU” problem.

5. Implementation Details

Experiments were conducted using PyTorch and HuggingFace Transformers. All feature extraction is performed on frozen SVELA language models, ensuring that verification remains strictly model-agnostic and does not modify parameters established during unlearning (Figure 1).

The official SVELA training split is internally divided using stratified sampling with a validation fraction of 20%, preserving class distribution across *retain*, *forget*, and *test*. Random seeds are fixed for reproducibility. Dimensionality reduction (IPCA) and feature standardization are fitted exclusively on the training subset and reused for validation and test splits to prevent information leakage.

The residual classifier is trained for 40 epochs using the AdamW optimizer with learning rate 3×10^{-4}

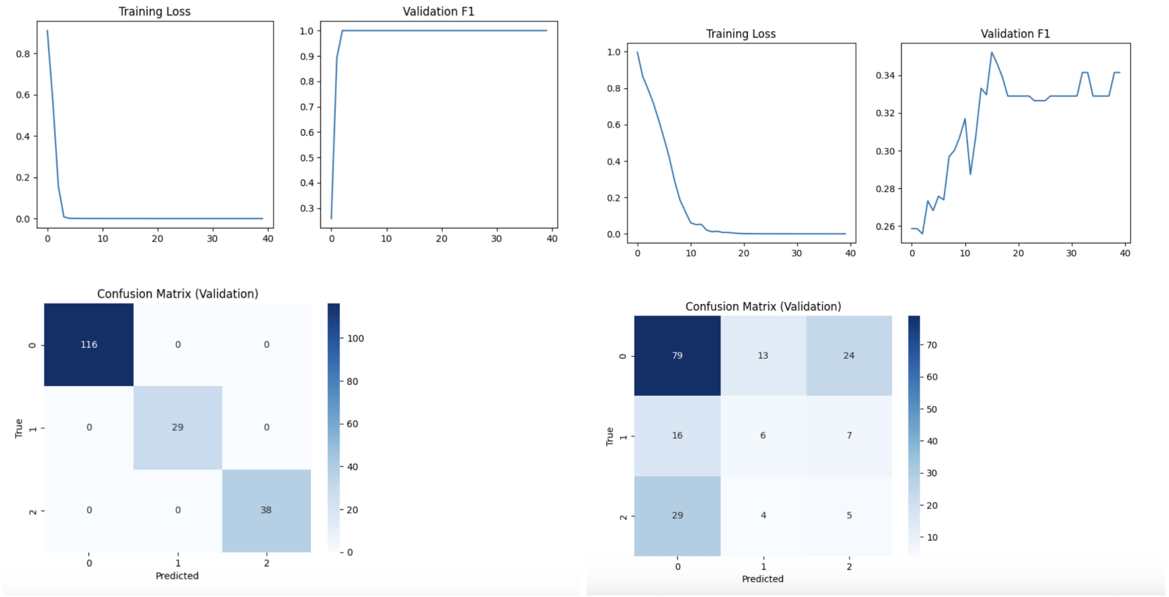


Figure 4: Training and validation performance. **Left**: Task 1 achieves nearly perfect separation within few epochs, reflecting its more distinct topic-based structure. **Right**: Task 2 shows slower convergence with class confusions, particularly between *retain* and *test*.

Table 4
Global hyperparameters shared across Task 1 and Task 2.

Parameter	Value
Optimizer	AdamW
Learning rate	3×10^{-4}
Loss function	Cross-Entropy
Classifier batch size	32
Hidden dimension	256
Number of residual blocks	2
Activation function	GELU
Dropout rate	0.1
Validation split	20% (stratified)
Random seed	42
Feature scaling	StandardScaler (train-only fit)

and batch size 32. The objective function is the multi-class cross-entropy loss:

$$\mathscr{L} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{3} \mathbb{1}[y_i = c] \log \hat{p}_{i,c}, \tag{5}$$

where y_i denotes the ground-truth label and $\hat{p}_{i,c}$ represents the predicted probability for class c. Model selection is performed using macro-averaged F1 score on the validation set. No explicit class weighting is applied, although the dataset is imbalanced in favour of the majority "retain" class in order to avoid information loss.

Global optimization and architectural hyperparameters shared across both tasks are summarized in Table 4, while task-specific settings are reported in Table 5.

Figure 4 provides insights into the training behavior and classification performance of the residual network for Task 1 and Task 2. Task 1 exhibits near-perfect validation F1 after just a few epochs, and the confusion matrix confirms excellent class separation. In contrast, in Task 2, the validation F1 improves gradually before plateauing, and the confusion matrix indicates frequent misclassification between the *retain* and *test* classes, reflecting the subtle semantic differences in that setup. This suggests that Task 1,

Table 5
Task-specific hyperparameters for Task 1 and Task 2.

Parameter	Task 1	Task 2
Max sequence length	384	512
IPCA components	128	200
Token batch size (feature extraction)	8	8
Training epochs	40	40

Table 6
Task 1 leaderboard results (mean and standard deviation of macro-averaged F1 score).

	1B Models		3B Models	
Team	Mean	Std	Mean	Std
priyam_saha17	**0.3428**	0.0100	**0.3565**	0.0068
MALTO	0.3363	0.0091	0.3526	0.0083
ItaLib	0.3184	0.0054	0.3109	0.0037
matteoberta	0.3183	0.0365	0.3284	0.0437

Table 7
Task 2 leaderboard results (mean and standard deviation of macro-averaged F1 score).

	1B Models		3B Models	
Team	Mean	Std	Mean	Std
priyam_saha17	**0.3345**	0.0045	**0.3335**	0.0049
MALTO	0.3234	0.0076	0.3321	0.0049
matteoberta	0.2882	0.0131	0.2838	0.0047

which includes entity-level forgetting, provides clearer discriminative signals for the classifier, likely due to more distinct shifts in representation between retained and forgotten instances.

6. Results

The system ranked first on all official leaderboards across both Task 1 and Task 2, for both 1B and 3B model settings. Tables 6 and 7 report the aggregate mean macro-averaged F1 scores (with standard deviation) across all evaluated unlearning algorithms. Across both tasks, the approach achieves the highest average macro-F1 compared to other participating teams. Performance remains competitive across model sizes, with slight variations between the 1B and 3B settings depending on the task and unlearning strategy.

Although submissions were made on four selected base models (Section 4.1), the evaluation scripts were executed on both 1B and 3B parameter variants under all considered unlearning strategies: adversarial negative training (advneg), standard fine-tuning on the retain set (finetune), KL-divergence minimization (kl_min), negative gradient unlearning (neggrad), and preference optimization (pref_opt).

The detailed class-wise analysis (Tables 8 and 9) reveals consistent disparities across classes. Class 0 ("Retain") generally achieves substantially higher precision, recall, and F1 scores compared to Classes 1 ("Forget") and 2 ("Test") across tasks, model sizes, and unlearning methods. In contrast, Classes 1 and 2 exhibit lower recall and F1 values, with performance varying more noticeably across unlearning strategies. This pattern suggests that the models retain stronger predictive performance for one class while exhibiting reduced separability or higher confusion for the remaining classes.

Table 8
Detailed results (Part 1): Overall score and class-wise accuracy for Task 1 and Task 2 (1B and 3B models).

Team	Task	Dim	Unlearner	Score	OverallAcc	C0Acc	C1Acc	C2Acc
priyam_saha17	task1	1b	advneg	0.3517	0.5100	0.6932	0.1528	0.2072
priyam_saha17	task1	1b	finetune	0.3540	0.5206	0.7143	0.1542	0.1910
priyam_saha17	task1	1b	kl_min	0.3379	0.4957	0.6780	0.1472	0.1887
priyam_saha17	task1	1b	neggrad	0.3300	0.4993	0.6932	0.1236	0.1771
priyam_saha17	task1	1b	pref_opt	0.3405	0.5084	0.7030	0.1458	0.1725
priyam_saha17	task1	3b	advneg	0.3508	0.5170	0.7073	0.1319	0.2141
priyam_saha17	task1	3b	finetune	0.3552	0.5138	0.6914	0.1208	0.2593
priyam_saha17	task1	3b	kl_min	0.3507	0.4941	0.6575	0.1514	0.2442
priyam_saha17	task1	3b	neggrad	0.3583	0.5138	0.6953	0.1750	0.2014
priyam_saha17	task1	3b	pref_opt	0.3673	0.5356	0.7295	0.1458	0.2245
priyam_saha17	task2	1b	advneg	0.3318	0.5035	0.6904	0.1264	0.1934
priyam_saha17	task2	1b	finetune	0.3345	0.5094	0.7018	0.1323	0.1934
priyam_saha17	task2	1b	kl_min	0.3253	0.4894	0.6680	0.1264	0.1812
priyam_saha17	task2	1b	neggrad	0.3234	0.4888	0.6737	0.1205	0.1763
priyam_saha17	task2	1b	pref_opt	0.3276	0.4941	0.6823	0.1264	0.1837
priyam_saha17	task2	3b	advneg	0.3299	0.5059	0.6932	0.1215	0.1837
priyam_saha17	task2	3b	finetune	0.3335	0.5100	0.7018	0.1323	0.1887
priyam_saha17	task2	3b	kl_min	0.3268	0.4894	0.6680	0.1323	0.1837
priyam_saha17	task2	3b	neggrad	0.3302	0.4976	0.6842	0.1264	0.1887
priyam_saha17	task2	3b	pref_opt	0.3321	0.5012	0.6904	0.1323	0.1887

Table 9
Detailed results (Part 2): Class-wise precision, recall and F1 scores for Task 1 and Task 2 (1B and 3B models).

				Class 0			Class 1			Class 2		
Team	Task	Dim	Unlearner	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
priyam_saha17	task1	1b	advneg	0.6417	0.6932	0.6664	0.2249	0.1528	0.1820	0.2062	0.2072	0.2067
priyam_saha17	task1	1b	finetune	0.6455	0.7143	0.6782	0.2418	0.1542	0.1883	0.2005	0.1910	0.1956
priyam_saha17	task1	1b	kl_min	0.6368	0.6780	0.6567	0.1924	0.1472	0.1668	0.1918	0.1887	0.1902
priyam_saha17	task1	1b	neggrad	0.6388	0.6932	0.6649	0.1784	0.1236	0.1460	0.1813	0.1771	0.1792
priyam_saha17	task1	1b	pref_opt	0.6396	0.7030	0.6698	0.2201	0.1458	0.1754	0.1804	0.1725	0.1763
priyam_saha17	task1	3b	advneg	0.6493	0.7073	0.6770	0.2317	0.1319	0.1681	0.2009	0.2141	0.2073
priyam_saha17	task1	3b	finetune	0.6509	0.6914	0.6705	0.2308	0.1208	0.1586	0.2173	0.2593	0.2364
priyam_saha17	task1	3b	kl_min	0.6501	0.6575	0.6538	0.2084	0.1514	0.1754	0.2051	0.2442	0.2229
priyam_saha17	task1	3b	neggrad	0.6513	0.6953	0.6726	0.2291	0.1750	0.1984	0.2064	0.2014	0.2039
priyam_saha17	task1	3b	pref_opt	0.6641	0.7295	0.6953	0.2574	0.1458	0.1862	0.2163	0.2245	0.2203
priyam_saha17	task2	1b	advneg	0.6378	0.6904	0.6630	0.2075	0.1264	0.1574	0.1872	0.1934	0.1902
priyam_saha17	task2	1b	finetune	0.6443	0.7018	0.6718	0.2137	0.1323	0.1635	0.1887	0.1934	0.1910
priyam_saha17	task2	1b	kl_min	0.6322	0.6680	0.6496	0.2050	0.1264	0.1563	0.1784	0.1812	0.1798
priyam_saha17	task2	1b	neggrad	0.6357	0.6737	0.6542	0.2018	0.1205	0.1508	0.1756	0.1763	0.1760
priyam_saha17	task2	1b	pref_opt	0.6395	0.6823	0.6602	0.2094	0.1264	0.1579	0.1816	0.1837	0.1826
priyam_saha17	task2	3b	advneg	0.6412	0.6932	0.6661	0.2049	0.1215	0.1527	0.1835	0.1837	0.1836
priyam_saha17	task2	3b	finetune	0.6468	0.7018	0.6731	0.2145	0.1323	0.1637	0.1879	0.1887	0.1883
priyam_saha17	task2	3b	kl_min	0.6339	0.6680	0.6505	0.2086	0.1323	0.1617	0.1795	0.1837	0.1816
priyam_saha17	task2	3b	neggrad	0.6381	0.6842	0.6603	0.2107	0.1264	0.1580	0.1834	0.1887	0.1860
priyam_saha17	task2	3b	pref_opt	0.6426	0.6904	0.6656	0.2139	0.1323	0.1637	0.1868	0.1887	0.1877

7. Discussion

The strong results across both SVELA tasks show that combining embeddings with uncertainty-based features (like entropy and logit margin) works well for verifying forgetting. These different signals seem to capture complementary aspects of the model's behavior, and using them together gives better performance than using any one signal alone.

Looking at the confusion matrices (Figure 4), most errors happen between the *retain* and *test* classes. This suggests that when forgetting is imperfect, the retained examples may still share some patterns with forgotten ones. Task 2 showed more of this confusion, likely because it involved removing individual facts (topics), which creates more overlap in the remaining data. In contrast, Task 1 dealt with entire identities, which gave the model cleaner separation between classes.

Further, using a fixed English prompt across all languages helped reduce variation in model output, which was useful for this multilingual setting. This strategy might help other multilingual evaluation tasks where consistent behavior is important.

Finally, Table 9 clearly shows that F1 scores for the “forget” and “test” classes are significantly lower than for the “retain” class across all unlearning strategies in both Task 1 and Task 2 (both 1B and 3B models). This suggests the classifier tends to overpredict the retain class which is likely due to its higher frequency and more consistent representations post-unlearning.

8. Limitations

While the proposed system attained top-ranked performance across all SVELA tracks, the absolute scores remain modest, with macro F1 hovering around 33–35%. This outcome highlights the inherent difficulty of selective forgetting verification and suggests that substantial headroom remains for future improvements.

The approach depends on access to labeled verification data, which may not be available in real-world unlearning deployments. Although the classifier is trained independently of the LLM parameters, the overall method still assumes the availability of structured prompt–response pairs and class annotations.

The system adopts several architectural choices that present opportunities for deeper exploration. For example, a shallow residual classifier is used with the assumption that residual connections support better gradient flow and feature fusion. While this design choice proved effective, comparative ablations with simpler MLP architectures were not conducted and could provide further clarity. Similarly, the dimensionality reduction stage relies on IPCA with empirically chosen component sizes based on validation performance; exploring alternative advanced methods such as autoencoders or manifold learning could offer additional benefits. As future work, advanced classifier architectures and richer feature extraction strategies may be incorporated to enhance generalization and interpretability across tasks.

Lastly, while class imbalance is handled by avoiding explicit downsampling of the “retain” class, the classifier displays a notable prediction bias towards this dominant class. The per-class F1 breakdown reveals persistent underperformance on the “forget” and “test” categories, which suggests that detecting forgotten content remains a challenging open problem.

9. Conclusion

This work presented a feature-based pipeline for verifying whether large language models have effectively forgotten targeted information following unlearning. The system relies on black-box access to LLMs and combines semantic embeddings with uncertainty and behavioral signals to train a lightweight classifier. All components are modular and task-agnostic, allowing the approach to generalize well across both entity-level and fact-level forgetting.

The method demonstrated consistent first-place performance across all four SVELA tracks, indicating the effectiveness of using fused representational and uncertainty features for this verification task. Despite its simplicity, the approach proved competitive and scalable across model sizes.

Future work could investigate unsupervised or semi-supervised strategies for verification, especially in settings where labeled forgotten examples are unavailable. Further ablation studies are also necessary to better understand the contribution of each component, including the effect of residual architecture and the role of individual uncertainty features. Broader evaluation across more diverse languages and unlearning algorithms could help stress-test the pipeline’s robustness and inform next-generation verification protocols.

Acknowledgments

The author thanks the SVELA organizers for providing the datasets, pretrained models, and evaluation infrastructure, and the EVALITA 2026 committee for organizing the shared task.

Declaration on Generative AI

During the preparation of this work, the author used GPT-5.2 and Grammarly to assist with sentence-level grammar correction and stylistic polishing of the manuscript, and figma for building the flowchart representation of the pipeline. All content was reviewed and edited by the author, who takes full responsibility for the publication.

References

[1] A. Ginart, M. Guan, G. Valiant, J. Zou, Making ai forget you: Data deletion in machine learning, NeurIPS (2019).

[2] L. Bourtoule, V. Chandrasekaran, C. A. Choquette-Choo, R. Jia, A. Travers, B. Zhang, D. Lie, N. Papernot, Machine unlearning, in: IEEE Symposium on Security and Privacy, 2021.

[3] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, Ú. Erlingsson, et al., Extracting training data from large language models, in: USENIX Security Symposium, 2021.

[4] D. Choi, D. Na, Towards machine unlearning benchmarks: Forgetting the personal identities in facial recognition systems, 2023. URL: https://arxiv.org/abs/2311.02240. arXiv:2311.02240.

[5] P. Maini, Z. Feng, A. Schwarzschild, Z. C. Lipton, J. Z. Kolter, Tofu: A task of fictitious unlearning for llms, 2024. URL: https://arxiv.org/abs/2401.06121. arXiv:2401.06121.

[6] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, C. Finn, Direct preference optimization: Your language model is secretly a reward model, 2024. URL: https://arxiv.org/abs/2305.18290. arXiv:2305.18290.

[7] J. Hayes, I. Shumailov, E. Triantafillou, A. Khalifa, N. Papernot, Inexact unlearning needs more careful evaluations to avoid a false sense of privacy, 2024. URL: https://arxiv.org/abs/2403.01218. arXiv:2403.01218.

[8] C. Savelli, M. La Quatra, A. Koudounas, F. Giobergia, Svela at evalita 2026: Overview of the selective verification of erasure from llm answers task, in: Proceedings of the Ninth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2026), CEUR.org, Bari, Italy, 2026.

[9] C. Savelli, M. La Quatra, A. Koudounas, F. Giobergia, FAME: Fictional actors for multilingual erasure, in: Proceedings of the Fifteenth Language Resources and Evaluation Conference, European Language Resources Association, 2026.

[10] F. Cutugno, A. Miaschi, A. P. Aprosio, G. Rambelli, L. Siciliani, M. A. Stranisci, Evalita 2026: Overview of the 9th evaluation campaign of natural language processing and speech tools for italian, in: Proceedings of the Ninth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2026), CEUR.org, Bari, Italy, 2026.

[11] D. Hendrycks, K. Gimpel, A baseline for detecting misclassified and out-of-distribution examples in neural networks, in: International Conference on Learning Representations, 2017.

[12] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, 2015. URL: https://arxiv.org/abs/1512.03385. arXiv:1512.03385.

Code and Reproducibility

To support transparency and reproducibility, the full implementation, training pipeline, and submission files used for both SVELA tasks are publicly available as Kaggle Notebooks:

- Task 1 notebook: https://www.kaggle.com/code/priyamsaha17/svela-task-1

- Task 2 notebook: https://www.kaggle.com/code/priyamsaha17/svela-task-2

A comprehensive GitHub repository containing all notebooks, trained models, evaluation scores, and performance curves is publicly available.[1]

[1] https://github.com/priyam-saha-17/A-Feature-Centric-Pipeline-for-Verifying-Selective-Forgetting-in-Large-Language-Models