UNLEARN
Efficient Removal of Knowledge in Large Language Models

Tyler Lizzo, Larry Heck
AI Virtual Assistant (AVA) Lab
Georgia Institute of Technology
{lizzo,larryheck}@gatech.edu

Abstract

Large Language Models (LLMs) excel in many Natural Language Processing tasks but are outperformed by specialized tools for certain tasks. This raises the question: Can we reduce redundant LLM parameters when using these tools? Given the size and high training costs of LLMs, it is essential to efficiently forget specific knowledge without retraining. This paper introduces UNLEARN, a novel method that uses subspace techniques to selectively remove knowledge without access to the original training data, without retraining, and with minimal impact to other tasks. Our results show that UNLEARN significantly outperforms previous methods for forgetting targeted (unwanted) knowledge while also preserving related (wanted) knowledge. We also propose LEARN, a complementary approach for targeted knowledge addition, which achieves fine-tuning accuracy comparable to Low-Rank Adaptation (LoRA) without degrading related task performance.[1]

1 Introduction

In recent years, Large Language Models (LLMs) have transitioned rapidly from research settings to practical applications, serving millions of users across diverse industries. Despite their broad versatility in natural language processing (NLP), LLMs face significant challenges in handling specific tasks such as arithmetic computation and causal reasoning, where simpler or more specialized task-specific tools often outperform them in terms of efficiency and accuracy. For instance, the Toolformer framework (Schick et al., 2023) demonstrates how specialized queries can be routed outside the LLM to external tools. This raises an important question: Can we eliminate or reduce the parameters within LLMs that are dedicated to these specialized tasks, which become redundant in a Toolformer-like architecture? Addressing this issue could lead to more efficient parameter utilization and reduced computational overhead by removing unnecessary task knowledge from the model.

[1]Code will be released at https://github.com/tylerlizzo/UNLEARN.

Current training paradigms offer limited solutions for addressing this inefficiency. One possible approach involves associating training samples with specific tasks and retraining the model to exclude redundant parameters. However, for modern LLMs with their immense scale, retraining the entire model is prohibitively expensive and time-consuming, rendering such approaches impractical.

This paper introduces UNLEARN, a novel algorithm that can forget or unlearn knowledge within an LLM without access to the original training data, without retraining, and without adversely affecting related knowledge. UNLEARN leverages subspace techniques to identify the subspaces spanned by particular knowledge (tasks) and discrimination methods to separate that subspace from subspaces of similar tasks. This allows the algorithm to prevent performance degradation when there are similar tasks, a common issue with traditional methods. Further, this technique uses a unified set of operators, where the task matrices are identical and used to either enhance or reduce the model's performance for a given task.

UNLEARN achieves 96% forgetting on the task of interest while maintaining performance on dissimilar tasks within 2.5% of the original model. When the tasks are similar, UNLEARN still achieves nearly 91% forgetting on the task of interest while preserving performance on similar tasks within 11%. These results significantly outperform the state-of-the-art, which achieves similar forgetting but is accompanied by significant degradation on similar tasks.

The forgetting of UNLEARN can easily be converted to *add knowledge* to the LLM. This new method LEARN matches the fine-tuning accuracy of the LoRA method (Hu et al., 2021) without affecting related tasks, demonstrating its dual nature across both knowledge unlearning and fine-tuning scenarios.

The contributions of this work are as follows:

- An efficient method to identify the subspace of specific knowledge within an LLM.

Findings of the Association for Computational Linguistics: NAACL 2025, pages 7272–7283
April 29 - May 4, 2025 ©2025 Association for Computational Linguistics

- A novel approach called subspace discrimination and task removal to selectively target and remove specific knowledge without adversely affecting other knowledge in the LLM.
- The introduction of LEARN, a dual algorithm to UNLEARN that provides a new approach to adding new knowledge to the LLM without affecting its other knowledge.

This paper presents the UNLEARN algorithm and demonstrates its performance in removing knowledge represented as tasks. Section 2 reviews the literature on Parameter Efficient Fine-Tuning, Machine Unlearning, and LLM Unlearning. Section 3 describes the three main parts of UNLEARN: subspace identification, subspace discrimination, and task removal. In Section 4, the performance of UNLEARN is tested over a large set of metrics and settings and compared to the current state-of-the-art. Section 4.5 introduces LEARN, a dual application of the UNLEARN algorithm for adding knowledge to the LLM. A comparison to traditional fine-tuning methods is made in Section 5. Future works are discussed in Section 6. Finally, Section 7 concludes the paper and outlines potential directions for future research.

2 Related Works

2.1 Parameter Efficient Fine-Tuning

Parameter Efficient Fine-Tuning (PEFT) is used to fine-tune large models without modifying most of the original pre-trained weights, resulting in significant computational and storage savings.

One of the most significant PEFT methods is Low-Rank Adaptation (LoRA; Hu et al., 2021), which decomposes weight updates into two low-rank matrices. While reducing trainable parameters by 10,000 times and GPU memory usage by 3 times, LoRA is still able to maintain the fine-tuning performance of a systems. Quantized Low-Rank Adaptation would build upon LoRA's performance gains by quantizing model weights (Dettmers et al., 2023).

Other notable PEFT methods include prompt tuning (Lester et al., 2021; Qin and Eisner, 2021), tuning hidden states (IA^3; Liu et al., 2022a), adding layers (Houlsby et al., 2019), tuning the embedding layer inputs (An et al., 2022), and hybrid approaches (Mahabadi et al., 2021). These extend prior work on domain adaptation of deep neural networks for Natural Language Processing (Jaech et al., 2016).

2.2 Machine Unlearning

Machine unlearning is the process of removing the influence of data on an already trained model, creating a model that behaves as if it was never trained on that data (Xu et al., 2023). While originally motivated by data protection regulations, such as the *California Consumer Privacy Act* (CCPA; Goldman, 2020) and the European Union's *General Data Protection Regulation* (GDPR; Goddard, 2017), unlearning has grown in relevance as models become more resource-intensive and the need for efficient domain removal has emerged.

Machine unlearning has since been extended to myriad areas: federated learning (Liu et al., 2022b; Zhang et al., 2023b), image classification (Bourtoule et al., 2021; Gupta et al., 2021; Liu et al., 2024a), and image generation (Gandikota et al., 2023; Kumari et al., 2023; Fan et al., 2024).

The most rigorous method for machine unlearning is 'exact' unlearning, completely retraining a model with the data points of interest removed (Yan et al., 2022; Nguyen et al., 2022; Fan et al., 2024). Although exact unlearning guarantees the removal of data, it is impractical for models of any significant size due to the high computation cost. For instance, training Llama-2-70B took $\sim$ 1.7 million GPU-hours on Nvidia A100 GPUs (Touvron et al., 2023).

2.3 LLM Unlearning

There is an increasing interest in machine unlearning in the context of LLMs (Jang et al., 2022; Meng et al., 2023; Liu et al., 2024c). Recent works highlight the importance of selective LLM unlearning to improve parameter efficiency and model adaptability.(Zhang et al., 2023a; Liu et al., 2024b; Schick et al., 2023).

Current methods for LLM unlearning include gradient ascent to reverse the learning of knowledge (Jang et al., 2022; Chen and Yang, 2023; Yao et al., 2024), preference optimization using alternative responses (Eldan and Russinovich, 2023; Maini et al., 2024), and input-based approaches (Pawelczyk et al., 2024; Thaker et al., 2024).

However, these methods face significant challenges. There are the aforementioned cost and time restraints. The vast amounts of training data used for LLM training adds to the complexity, as identifying and isolating the specific data points to be unlearned is a non-trivial task (Eldan and Russinovich, 2023; Ilharco et al., 2023). The scope of unlearning is generally under-specified; unlearning should remove knowledge within the scope of the targeted data while maintaining performance on other data (Mitchell et al., 2022). Finally, there is a lack of comprehensive evaluation methods to assess the effectiveness of unlearning in LLMs (Patil et al., 2023; Shi et al., 2024).

2.4 Toolformer

Toolformer (Schick et al., 2023) addresses a crucial limitation in LLMs: while these models excel at complex language tasks, they often struggle with simpler tasks like arithmetic and factual lookup, where small, specialized systems perform better. This inefficiency underscores the need to offload such tasks from LLMs to more appropriate downstream tools. Toolformer enables LLMs to call external APIs for these tasks, resulting in superior performance compared to the base LLM. This approach closely aligns with the goal of enhancing parameter efficiency by offloading tasks and reducing the burden of unnecessary knowledge in LLMs, relying on specialized tools instead. This underscores the need for tools to offload knowledge better handled by external tools, ensuring the LLM focuses on tasks where it provides the most value.

3 UNLEARN Method

The method proposed in this paper consists of three main tasks: subspace identification, discrimination, and removal[2]. Subspace identification trains a knowledge (task)-dependent matrix for a specified layer while freezing all other layers. This sequential, layer-by-layer training starts with the first layer and progresses through the entire network to yield a set of matrices that represent the task-dependent subspace (Section 3.1). Once identified, subspace discrimination removes the information unique to the task of interest while preventing any degradation of other tasks. This is achieved using a variation of the Gram-Schmidt process to orthogonalize subspaces, allowing mutual information to be preserved (Section 3.2). The final step is subspace removal, where the modified task matrix, T_i', is subtracted (Section 3.3).

3.1 Subspace Identification

This step identifies the subspace of a specific task within the LLM weight space. The method utilizes a general training that is implemented layer-by-layer, starting with the first layer ($l = 1$). All training is performed with a train/validation/test split of 0.6/0.2/0.2: The train set is used for training the network, the validation set determines when to stop training for a specific layer in our sequential process, and all evaluations are performed on the final test set:

0. **Model:** The original pre-trained weights of the LLM are removed and the weights for all layers are randomly initialized.
1. **Layer Freezing:** Except for the weights at layer l, all other weights for the subsequent layers of an LLM are frozen to isolate the training to one layer at a time.
2. **Training:** Training is completed on the task dataset with the l-th layer unfrozen. This is achieved by maximizing the conditional language modeling objective:

$$\max_{T_i^l} \sum_{(x,y)\in\mathcal{Z}} \sum_{t=1}^{|y|} \log(P_{T_i^l}(y_t|x, y_{<t})) \quad (1)$$

   where x_i and y_i are sequences of tokens and $T_i^l \in \Re^{n\times n}$ is the matrix for task i at the l-th layer and $n \times n$ the dimensions of the original pre-trained weight matrix.

   Given the matrix T_i^l is trained on a specific task, the matrix is likely rank deficient. To facilitate training, we alter each layer using a bottleneck architecture as shown in Figure 2 with interior dimension k, where $T_i^l = FG$.
3. **Sequential Training:** Once the training at layer l is complete, that layer is frozen and the next layer is unfrozen. For our experiments, training concluded once loss on the validation set had stopped decreasing (i.e. potential overfitting of the training set was starting). Similar training is then performed on the next layer. This process is repeated across all layers, resulting in weight matrices for each layer.

By the end of this sequential training and freezing process, shown in Figure 1, the set of weight matrices captures an accurate representation of the task-dependent subspace within the weights of the Transformer model. This method is lightweight, maintaining the computational efficiency of low rank training. The layer-by-layer approach was taken because the early layers contain higher-level semantic information, while the later layers contain more task/fact-specific information. Training

[2]The code will be released with the camera-ready version under GNU Public License.

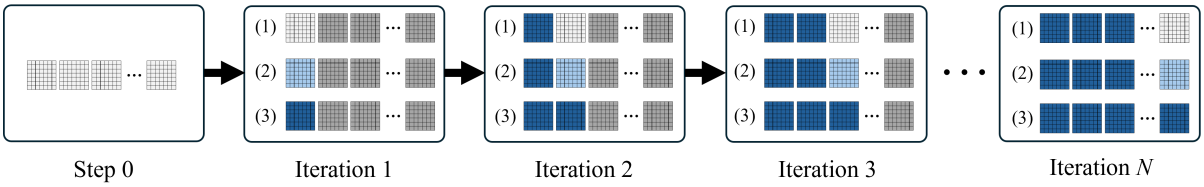

Figure 1: The Subspace Identification Process. The process begins by randomly initializing the model weights and then freezing them. Then an iterative process of unfreezing, training, and refreezing each layer occurs. This results in a set of matrices that capture an accurate representation for that task.

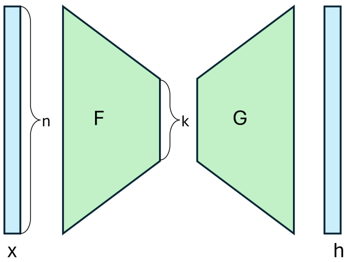

Figure 2: Bottleneck architecture of layer l with interior dimension $k \ll n$

in this method ensures the most reliable identification of the tasks.

3.2 Subspace Discrimination

Once a task-dependent subspace has been identified, it could be removed by subtracting it from the entire weight space (layer-by-layer). While this may be effective at removing the task of interest, it leads to performance loss when similar tasks are also evaluated, i.e. ones that occupy similar subspaces. Therefore, a method is required that maintains the mutual information between these two subspaces, only removing the information unique to the task of interest. We call this subspace discrimination.

To achieve subspace discrimination, we utilize a variation of the Gram-Schmidt process. Gram-Schmidt is used to orthogonalize a set of vectors in an inner product space. Given the subspace U spanned by vectors $u_1, \cdots, u_N$, we can find the orthogonal subspace to a vector v_k with the following:

$$v'_k = v_k - \sum_{j=1}^{N} \frac{\langle v_k, u_j \rangle}{\langle u_j, u_j \rangle} u_j.$$

A proof that v'_k is orthogonal to all u_j is offered in Appendix A. For our application, we compute:

$$SV_k(T'_i) = SV_k(T_i) - \sum_{j=1}^{N} \frac{SV_k(T_i) \cdot SV_j(T_o)}{SV_j(T_o) \cdot SV_j(T_o)} SV_j(T_o)$$

where T_i represents the identified subspace to be removed, T_o represents a similar task, and $SV_k(T'_i)$ represents the k-th singular vector of matrix T_i for one of the Transformer layers l. When applied to two tasks, every pair of weight matrices is decomposed and separated in this manner. For three or more tasks, the other task matrices; $T_{o,1}$, $T_{o,2}, \cdots$, $T_{o,n}$, are added into one T_o matrix, then the above equation is applied. We chose to use Euclidean inner products, inspired by the original LoRA paper (Hu et al., 2021), which demonstrated that efficient training could be achieved with linear rank decompositions. While neural network parameter spaces are non-Euclidean, the practical success of the LoRA method justified our approach.

Initially, the similarity of tasks was determined subjectively. However, this subspace discrimination method allows us to quantify task similarity, as there will be more overlap in the weight space of two similar sets of matrices. For two dissimilar tasks, the discrimination process will have no effect, as they are already orthogonal.

Subspace discrimination is essential to the UNLEARN algorithm, allowing for the precise separation of task-specific information within shared weight spaces and ensuring that the removal of one task does not undesirably impact the performance on similar tasks. Consequently, subspace discrimination enhances the algorithm's adaptability and robustness.

3.3 Task Removal

The final step removes the task subspace. To achieve this, our approach uses SVD reconstruction to reconstitute the modified task matrix, T'_i from the singular values of T_i and singular vectors $SV(T'_i)$ above. We can directly subtract the modified task T'_i, from W', or, more generally, subtract a linear smoothing of the task subspaces T'_i and T_i

$$W' = W - \alpha T'_i - (1 - \alpha) T_i$$

where $\alpha \in [0, 1]$ governs the relative strength of the two UNLEARN matrices (with and without discrimination). By including the smoothing factor, we can balance the impact of removal on the targeted task while mitigating unwanted degradation on the similar task.

4 Experiments

All experiments in this section use the same setup, with Llama-2-70b serving as the LLM. For the training step in the subspace identification method (Section 3.1) as illustrated in Figure 2, we used the Python package LoRALib (Hu et al., 2021) but, rather than training a fine-tuning adapter, we modified it to train the bottleneck in Figure 2 from scratch. We used a rank of $k = 16$. Only the attention matrices were modified during training. This was inspired by the original LoRA paper (Hu et al., 2021), where they only adapted the attention weights.

4.1 Datasets

A diverse selection of benchmarks is essential to evaluate performance degradation across similar tasks when modifying task-specific subspaces within LLMs. This study used two signification collections of benchmarks: Holistic Evaluation of Language Models (HELM; et al., 2023c) and the Beyond-the-Imitation-Game Benchmark (BIG-Bench; et al., 2023a).

HELM evaluates a wide range of use cases and metrics, encompassing general language abilities to simple question-answering settings. This benchmark evaluates models across multiple metrics–accuracy, fairness, robustness, efficiency, and more–providing a detailed view into the general language capabilities of models.

Complementing HELM, BIG-Bench focuses on more specific and niche tasks that probe the boundaries of current LLM capabilities. With 204 tasks contributed from experts across fields, BIG-Bench was invaluable for testing specific tasks that were beyond the domain of HELM. Importantly, BIG-Bench provided niche tasks that have little overlap with other tasks, offering an unbiased perspective on subspace removal.

Together, these datasets facilitate a comprehensive analysis of the influence of subspace removal on LLM performance across a spectrum of tasks. By integrating the thorough evaluation of HELM for general language abilities with the specialized tasks from BIG-Bench, this study explores how manipulation of tasks affects both broad and targeted model capabilities. This sheds light on the ability of UNLEARN to remove a task without affecting adjacent tasks.

4.2 Task Removal without Discrimination

The first experiment evaluated the UNLEARN method using only subspace identification (Section 3.1) and task removal without the subspace discrimination method ($\alpha = 0$ in Section 3.3). In these experiments, a single task was removed and performance across a set of tasks was observed.

Table 1 shows the performance of UNLEARN where the math word problem dataset GSM8K (Cobbe et al., 2021) was removed. Referring to the row with $\alpha = 0$, the first six evaluation tasks ranging from question-answering (NarrativeQA; Kocisky et al., 2017) to more general benchmarks like (MMLU; Hendrycks et al., 2021) were chosen because they are very different tasks from GSM8K. Because these tasks are dissimilar, they theoretically have little overlap in their weight subspaces. Evaluating the six chosen benchmarks on both the base model and UNLEARN ($\alpha = 0$) shows our approach successfully forgets (dropped performance) by 96.5% on the desired GSM8K task. In contrast, all other tasks had minimal degradation (less than 2.5%).

Referring to the last column of Table 1, an additional benchmark was added called arithmetic from BIG-Bench. The addition of this task examines the performance of UNLEARN when the goal is to remove one task, GSM8K, while preserving a highly related task, arithmetic. In this case, while UNLEARN without task discrimination ($\alpha = 0$) successfully removed the targeted task, GSM8k, by 96%, it also adversely affected the arithmetic task (down 33%).

This outcome underscores the challenges with task-specific subspace removal when dealing with closely aligned tasks. The performance decline on the second task suggests that the extracted subspace on the first task contains features shared by the second's subspace, highlighting the need for the subspace discrimination technique of Section 3.2.

4.3 Task Removal with Discrimination

The last row of Table 1 with $\alpha = 1$ corresponds to the UNLEARN method with the Task Discrimination method of Section 3.2. With GSM8K as the targeted task to be removed, the knowledge from

Table 1: Performance of UNLEARN when targeting GSM8K for removal and preserving NarrativeQA (NQA), NaturalQuestions (NQ), Massive Multitask Language Understanding (MMLU), IMDB sentiment analysis in movies (IMDB), Real-world Annotated Few-Shot (RAFT), Grade School Math 8K (GSM8K), and arithmetic.

α	Evaluation Tasks						
	NQA	NQ	MMLU	IMDB	RAFT	GSM8K	arithmetic
Base Model	0.778	0.680	0.583	0.952	0.719	0.483	0.991
0	0.758	0.681	0.577	0.949	0.715	0.017	0.633
0.25	0.755	0.681	0.566	0.951	0.712	0.041	0.692
0.5	0.768	0.670	0.571	0.932	0.706	0.046	0.781
0.75	0.749	0.664	0.579	0.946	0.708	0.045	0.878
1	0.772	0.674	0.582	0.946	0.723	0.087	0.956

the six unrelated tasks (first six) was once again preserved with a reduction in the GSM8K task by 82%. While the removal of the targeted task was not as pronounced when using the task discrimination method, the related arithmetic task was much less adversely affected with only a 3.5% reduction versus 33% when $\alpha = 0$.

To explore the dynamics of the subspace discrimination process, we varied the smoothing factor α introduced in Section 3.3. Again, GSM8K was the targeted task to remove. As shown in Table 1, while we see forgetting of GSM8K degrade as α increases, the preservation of the adjacent task, and arithmetic improves at a much faster rate. For example, with $\alpha = 0.75$ in the table and as shown in Figure 3, we find a balanced tradeoff with the UNLEARN method matching the best forgetting of GSM8K of previous methods, Knowledge Unlearning (KU), with a 91% reduction while significantly outperforming KU and the others in preserving the arithmetic task with only an 11% reduction (versus a 50% reduction with KU).

When arithmetic was the targeted task, the results had deleterious effects on GSM8K as well[3]. Varying values of α (i.e. making the discrimination process more aggresive) had the unintended effect of reducing the unlearning impact on both tasks. This suggests these tasks' subspaces entirely overlap, preventing the successful discrimination of the two.

4.4 Optimal Rank

We explore the impact of varying the rank of the rank-deficient matrices during subspace identification, as shown in Figure 2. As seen in Table 2 with the NQA as the targeted task to be removed, we vary the rank from k =1,2,4,8,16,32. The performance is not hindered for k values above 4 but

[3]Detailed results are shown in the Appendix in Table 4.

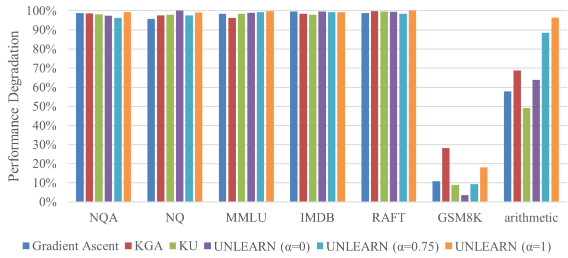

Figure 3: Performance degradation for each task when GSM8K is the targeted task. The plot shows the percentage of performance retained for each task across four different models compared to the base model: Gradient Ascent, Knowledge Gap Alignment (KGA), Knowledge Unlearning (KU), and UNLEARN ($\alpha = 0.75$).

there is a slight degradation of performance on the task of interest for the lower-rank experiments; the task of interest was not forgotten as effectively, and the similar tasks (NQ and MMLU) experienced greater performance degradation. This result can be attributed to the subspace identification step not capturing the subspaces for those tasks as accurately when the rank is lower.

These results suggest that the rank can be significantly reduced with minimal performance loss. This is reasonable given that the subspaces of interest were quite small compared to the overall dimensions of the weight matrices. In Llama-2-70B, with dimension $N = 5120$, removing a single task subspace would result in a 0.16% reduction in parameters, assuming an intrinsic rank of $k = 4$. Furthermore, this reduction scales almost linearly with the number of tasks removed, especially when the tasks are highly orthogonal. Reducing the parameter count by 10% only requires the removal of 66 orthogonal task subspaces. This adaptability makes the method highly suitable for resource-constrained environments, where minimizing the model's parameter count without sacrificing performance is critical.

Table 2: Performance of UNLEARN when the rank (k) is modified and the Targeted Task to be removed is Narrative QA (NQA). Detailed results are shown in the Appendix in Table 5.

k	Evaluation Tasks		
	NQA	NQ	MMLU
Base Model	0.778	0.68	0.583
1	0.167	0.599	0.58
2	0.151	0.609	0.564
4	0.128	0.624	0.568
8	0.136	0.627	0.58
16	0.135	0.628	0.581
32	0.134	0.619	0.579

4.5 Using UNLEARN to LEARN

4.5.1 LEARN methodology

The UNLEARN methodology, initially designed for the selective removal of task-specific information from LLMs, also presents a versatile framework that can be adapted for the enhancement of model performance on particular tasks. This section explores 'LEARN,' the application of our earlier UNLEARN algorithm for training on new information. This method aims to *add knowledge* and/or amplify the representation of a given task within the model, leading to improved performance on that task.

The LEARN approach uses the same principles as UNLEARN but inverts the application to focus on task enhancement. Specifically, the method involves identifying the subspace associated with a desired task using the approach in Section 3.1; this step is identical to UNLEARN. The difference comes with task addition instead of task removal; the only necessary change is flipping the equation for task removal from Section 3.3:

$$W' = W + T_i'$$

This addition should bolster performance on a new task, as the T_i' sits on top of the existing weight matrix, similar to the function of most LLM adapters. In addtion, due to subspace discrimination (Section 3.2), adding the new knowledge should have minimal adverse effects on other knowledge already in the LLM.

4.5.2 LEARN evaluation

To evaluate the effect of the LEARN method, experiments were conducted on tasks where pre-trained models showed suboptimal performance but had the potential to perform well if fine-tuned. Identifying tasks that meets these criteria for larger LLMs (50 B+ parameter) is challenging because they are trained on such extensive datasets that it is more difficult to find data not included in the training set. Therefore, by restricting the size of the LLM, we limit the total learning capacity of the model, allowing us to squeeze out additional learning that the LLM should be able to handle.

Table 3: Performance of LEARN and LoRA on LegalBench

Task	Base Model	LEARN	LoRA
Issue	50.1	73.4	72.9
Rule	42.7	61.8	63.1
Conclusion	53.9	69.3	69.6
Interpretation	48.1	68.1	67.4
Rhetorical	45.4	62.5	61.2
Average	48.0	67.0	66.8

These experiments used a similar setting to before, with the exception of using Llama-2-7b. The dataset of interest is LegalBench, a benchmark built by a collaboration between lawyers and ML engineers to measure legal reasoning in LLMs (et al., 2023b). Llama-2-7b performs between 30-50% across all tasks, leaving room for improvement.

When the LEARN algorithm was applied to the model for LegalBench, it showed marked improvement across all tasks. Table 3 shows the consistent improvement across tasks and a 40% boost to the average performance of the system compared to the base LLM. Training with LEARN is shown relative to traditional LoRA fine-tuning. Only the two tasks of interest were shown in Table 3 because there was a similar lack of impact on the other tasks. LEARN matches the performance of LoRA. By systematically adding task-specific subspaces, LEARN fine-tunes the model's performance on a selected task and minimizes any degradation of other capabilities due to the subspace discrimination method. The dual capability of UNLEARN/LEARN underscores its main value: the ability to use the same training runs for both forgetting and learning.

5 Comparison to Existing Methods

This section presents a comparative analysis of the UNLEARN/LEARN methodology against existing methods, with a focus on generality and task performance.

5.1 Generality and Efficiency

A key advantage of UNLEARN/LEARN is its operational flexibility. It offers a generalized framework that can be applied to full fine-tuning or any PEFT method for fine-tuning. UNLEARN/LEARN applies the same underlying principles in any setting–either adding or subtracting task-specific matrices from the model's weight matrices–to both enhance (LEARN) and diminish (UNLEARN) the model's performance on specific tasks. Because the same set of matrices are being used regardless of algorithm, this simplifies model management and reduces the computational and storage overhead.

5.2 Task Performance

In scenarios involving similar tasks, the differences between UNLEARN/LEARN and existing methods become even more pronounced. In the LEARN setting of Table 3, both methods show comparable improvements in task performance, demonstrating their efficacy for bolstering model performance. In the forgetting setting, the UNLEARN algorithm is able to successfully discriminate between two similar tasks and only remove the task of interest.

We compared UNLEARN to the current state-of-the-art algorithms: Gradient Ascent (Yao et al., 2024), Knowledge Gap Alignment (KGA; Wang et al., 2023), and Knowledge Unlearning (KU Jang et al., 2022). As seen in Table 4, these state-of-the-art methods are unable to discriminate effectively between tasks, leading to performance degradation in closely related tasks. For example, when NarrativeQA is the task of interest, UNLEARN successfully degrades that task (down from 0.778 to 0.135) while maintaing the performance on NaturalQuestions (from 0.680 to 0.628). All three state-of-the-art algorithms successfully degrade NarrativeQA: GA degrades the task to 0.094, KGA to 0.183, and KU to 0.163. However, they all show significantly diminished performance on NaturalQuestions: GA degrades the task to 0.415, KGA to 0.229, and KU to 0.329. These state-of-the-art methods lack the discrimination ability to target the knowledge they seek to remove without unwanted performance effects on secondary tasks.

Conversely, with its precise subspace manipulation, the UNLEARN method allows for the selective removal of task influences without negatively impacting the performance of related tasks. This specificity is particularly beneficial in multi-task learning/unlearning environments where tasks share overlapping features (similar weight subspaces). As such, UNLEARN is better suited for forgetting tasks while preserving similar tasks.

6 Future Works

This paper has laid the groundwork for several intriguing avenues for future research. First, while our initial work focused on removing broad domain knowledge, future efforts will extend this methodology to the removal of specific knowledge and facts. We are currently collecting datasets that will facilitate this extension, particularly in scenarios involving private or harmful information.

There are some scalability concerns if UNLEARN is applied to a large number of tasks. While the current work targets the selective removal of a small number of unwanted tasks, future research will investigate strategies to efficiently handle discrimination between larger sets of similar tasks.

Our current approach was largely inspired by the original LoRA paper (Hu et al., 2021), which was our motivation for only manipulating the attention weights. Subsequent research into LoRA revealed the effectiveness of manipulating the other layers within an LLM. Future works will explore the adaption of other layers to enhance the flexibility and performance of UNLEARN.

7 Conclusion

This paper introduces UNLEARN, a novel approach for selectively forgetting knowledge in Large Language Models. This method relies on subspace identification for tasks and subspace discrimination between similar tasks. Compared to state-of-the-art methods like Gradient Ascent, UNLEARN offers substantial advantages in terms of simplicity, generality, efficiency, precision, and overall effectiveness. The experimental results demonstrate significant performance gains, highlighting the effectiveness of UNLEARN in removing unwanted knowledge without causing significant degradation on related tasks that are not fully contained within the targeted task. The method's ability to accurately isolate and remove specific subspaces within the model ensures precise unlearning, making it a valuable tool for managing the complexities of task forgetting.

This paper also showed that UNLEARN can be easily reconfigured to *learn* new tasks. This complementary method, LEARN, is an approach for targeted knowledge addition that achieves fine-tuning accuracy comparable to Low-Rank Adaptation (LoRA) without degrading related task performance.

Limitations

Although UNLEARN enhances the abilities of LLMs to forget knowledge, certain limitations still need to be addressed. One limitation is when tasks completely overlap, as observed with arithmetic and GSM8K. When a subspace is entirely contained within another, as arithmetic was within GSM8K, it becomes challenging to discriminate between these two tasks. This highlights the distinction between knowledge and the metrics that measure knowledge, which we will explore this distinction in future works.

Another limitation of this paper that will be addressed in future work is to more fully leverage the experimental insights to optimize the efficiency of the UNLEARN method.

Ethics Statement

Acknowledgements

This work was supported by CoCoSys, one of seven centers in JUMP 2.0, a Semiconductor Research Corporation (SRC) program sponsored by DARPA.

References

Shengnan An, Yifei Li, Zeqi Lin, Qian Liu, Bei Chen, Qiang Fu, Weizhu Chen, Nanning Zheng, and Jian-Guang Lou. 2022. Input-tuning: Adapting unfamiliar inputs to frozen pretrained models.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2021. Machine unlearning. In 2021 IEEE Symposium on Security and Privacy (SP), pages 141–159.

Jiaao Chen and Diyi Yang. 2023. Unlearn what you want to forget: Efficient unlearning for llms.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms.

Ronen Eldan and Mark Russinovich. 2023. Who's harry potter? approximate unlearning in llms.

Aarohi Srivastava et al. 2023a. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models.

Neel Guha et al. 2023b. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models.

Percy Liang et al. 2023c. Holistic evaluation of language models.

Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. 2024. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation.

Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. 2023. Erasing concepts from diffusion models.

Michelle Goddard. 2017. The eu general data protection regulation (gdpr): European regulation that has a global impact. International Journal of Market Research, 59(6):703–705.

Eric Goldman. 2020. An introduction to the california consumer privacy act (ccpa). Santa Clara Univ. Legal Studies Research Paper.

Varun Gupta, Christopher Jung, Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi, and Chris Waites. 2021. Adaptive machine unlearning. In Advances in Neural Information Processing Systems, volume 34, pages 16319–16330. Curran Associates, Inc.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic.

Aaron Jaech, Larry Heck, and Mari Ostendorf. 2016. Domain adaptation of recurrent neural networks for natural language understanding. arXiv preprint arXiv:1604.00117.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. 2022. Knowledge unlearning for mitigating privacy risks in language models.

Tomas Kocisky, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2017. The narrativeqa reading comprehension challenge.

Nupur Kumari, Bingliang Zhang, Sheng-Yu Wang, Eli Shechtman, Richard Zhang, and Jun-Yan Zhu. 2023. Ablating concepts in text-to-image diffusion models.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022a. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning.

Mengda Liu, Guibo Luo, and Yuesheng Zhu. 2024a. Machine unlearning with affine hyperplane shifting and maintaining for image classification. In Neural Information Processing, pages 215–227, Singapore. Springer Nature Singapore.

Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Xiaojun Xu, Yuguang Yao, Hang Li, Kush R. Varshney, Mohit Bansal, Sanmi Koyejo, and Yang Liu. 2024b. Rethinking machine unlearning for large language models.

Yi Liu, Lei Xu, Xingliang Yuan, Cong Wang, and Bo Li. 2022b. The right to be forgotten in federated learning: An efficient realization with rapid retraining. In IEEE INFOCOM 2022 - IEEE Conference on Computer Communications, pages 1749–1758.

Zheyuan Liu, Guangyao Dou, Zhaoxuan Tan, Yijun Tian, and Meng Jiang. 2024c. Towards safer large language models through machine unlearning.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C. Lipton, and J. Zico Kolter. 2024. Tofu: A task of fictitious unlearning for llms.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2023. Locating and editing factual associations in gpt.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D. Manning, and Chelsea Finn. 2022. Memory-based model editing at scale.

Thanh Tam Nguyen, Thanh Trung Huynh, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and Quoc Viet Hung Nguyen. 2022. A survey of machine unlearning.

Vaidehi Patil, Peter Hase, and Mohit Bansal. 2023. Can sensitive information be deleted from llms? objectives for defending against extraction attacks.

Martin Pawelczyk, Seth Neel, and Himabindu Lakkaraju. 2024. In-context unlearning: Language models as few shot unlearners.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools.

Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2024. Detecting pretraining data from large language models.

Pratiksha Thaker, Yash Maurya, and Virginia Smith. 2024. Guardrail baselines for unlearning in llms.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Lingzhi Wang, Tong Chen, Wei Yuan, Xingshan Zeng, Kam-Fai Wong, and Hongzhi Yin. 2023. Kga: A general machine unlearning framework based on knowledge gap alignment.

Heng Xu, Tianqing Zhu, Lefeng Zhang, Wanlei Zhou, and Philip S. Yu. 2023. Machine unlearning: A survey.

Haonan Yan, Xiaoguang Li, Ziyao Guo, Hui Li, Fenghua Li, and Xiaodong Lin. 2022. Arcane: An efficient architecture for exact machine unlearning. In IJCAI, volume 6, page 19.

Yuanshun Yao, Xiaojun Xu, and Yang Liu. 2024. Large language model unlearning.

Dawen Zhang, Pamela Finckenberg-Broman, Thong Hoang, Shidong Pan, Zhenchang Xing, Mark Staples, and Xiwei Xu. 2023a. Right to be forgotten in the era of large language models: Implications, challenges, and solutions.

Lefeng Zhang, Tianqing Zhu, Haibin Zhang, Ping Xiong, and Wanlei Zhou. 2023b. Fedrecovery: Differentially private machine unlearning for federated learning frameworks. IEEE Transactions on Information Forensics and Security, 18:4732–4746.

A Proof of Orthogonality of v_k'

We offer a quick proof that v_k' is orthogonal to the orthogonal components of U: $u_1, \cdots, u_N$. We begin with our orthogonality definition in an inner product space:

$$u, v \text{ are orthogonal if } \langle u, v \rangle = 0$$

Next, we consider v_k' and arbitrary u_ℓ:

$$v_k' = v_k - \sum_{j=1}^{N} \frac{\langle v_k, u_j \rangle}{\langle u_j, u_j \rangle} u_j$$

We need to show that $\langle v_k', u_\ell \rangle = 0$. We proceed with the following calculation:

$$\begin{aligned}
\langle v_k', u_\ell \rangle &= \left\langle v_k - \sum_{j=1}^{N} \frac{\langle v_k, u_j \rangle}{\langle u_j, u_j \rangle} u_j, u_\ell \right\rangle \\
&= \langle v_k, u_\ell \rangle - \left\langle \sum_{j=1}^{N} \frac{\langle v_k, u_j \rangle}{\langle u_j, u_j \rangle} u_j, u_\ell \right\rangle \\
&= \langle v_k, u_\ell \rangle - \sum_{j=1}^{N} \frac{\langle v_k, u_j \rangle}{\langle u_j, u_j \rangle} \langle u_j, u_\ell \rangle
\end{aligned}$$

Since $u_1, \cdots, u_N$ are orthogonal components, we have $\langle u_j, u_k \rangle = 0$ for $j \neq k$. This simplifies the summmation as follows:

$$\begin{aligned}
\langle v_k', u_\ell \rangle &= \langle v_k, u_\ell \rangle - \sum_{j=1}^{N} \frac{\langle v_k, u_j \rangle}{\langle u_j, u_j \rangle} \langle u_j, u_\ell \rangle \\
&= \langle v_k, u_\ell \rangle - \frac{\langle v_k, u_\ell \rangle}{\langle u_\ell, u_\ell \rangle} \langle u_\ell, u_\ell \rangle \\
&= \langle v_k, u_\ell \rangle - \langle v_k, u_\ell \rangle \\
&= 0
\end{aligned}$$

Thus, we have shown that $\langle v_k', u_\ell \rangle = 0$ for any u_ℓ, proving that v_k' is orthogonal to the orthogonal components, $u_1, \cdots, u_N$, of U.

B Detailed Results of UNLEARN method

Full results are found in Table 4 below.

C Detailed Results of the Rank Reduction

Full results are found in Table 5 below.

Table 4: Performance of UNLEARN on a variety of tasks, compared to three state-of-the-art models: Gradient Ascent (Yao et al., 2024), Knowledge Gap Alignment (KGA, Wang et al., 2023), and Knowledge Unlearning (KU, Jang et al., 2022). Targeted Task represents the task that was 'unlearned'. The tasks of interest are NarrativeQA (NQA), NaturalQuestions (NQ), Massive Multitask Language Understanding (MMLU), IMDB benchmark for sentiment analysis in movies (IMDB), Real-world Annotated Few-Shot (RAFT), Grade School Math 8K (GSM8K), and arithmetic. The green columns represent the targeted task and the yellow columns represent the similar task.

Targeted Task	Model	Evaluation Tasks						
		NQA	NQ	MMLU	IMDB	RAFT	GSM8K	arithmetic
	Base Model	0.778	0.680	0.583	0.952	0.719	0.483	0.991
GSM8K	Gradient Ascent	0.768	0.651	0.574	0.949	0.710	0.052	0.574
	KGA	0.767	0.664	0.561	0.937	0.718	0.136	0.682
	KU	0.763	0.666	0.574	0.933	0.716	0.043	0.487
	UNLEARN ($\alpha = 0$)	0.758	0.681	0.577	0.949	0.715	0.017	0.633
	UNLEARN ($\alpha = 0.25$)	0.755	0.681	0.566	0.951	0.712	0.041	0.692
	UNLEARN ($\alpha = 0.5$)	0.768	0.670	0.571	0.932	0.706	0.046	0.781
	UNLEARN ($\alpha = 0.75$)	0.749	0.664	0.579	0.946	0.708	0.045	0.878
	UNLEARN ($\alpha = 1.0$)	0.772	0.674	0.582	0.946	0.723	0.087	0.956
arithmetic	Gradient Ascent	0.782	0.663	0.577	0.953	0.713	0.215	0.084
	KGA	0.767	0.675	0.581	0.939	0.700	0.105	0.017
	KU	0.760	0.672	0.567	0.942	0.719	0.183	0.063
	UNLEARN ($\alpha = 0$)	0.757	0.680	0.578	0.949	0.716	0.087	0.028
	UNLEARN ($\alpha = 0.25$)	0.771	0.673	0.584	0.949	0.717	0.214	0.229
	UNLEARN ($\alpha = 0.5$)	0.762	0.680	0.575	0.941	0.713	0.277	0.462
	UNLEARN ($\alpha = 0.75$)	0.773	0.676	0.571	0.948	0.709	0.363	0.628
	UNLEARN ($\alpha = 1.0$)	0.771	0.681	0.569	0.955	0.712	0.461	0.825
NQA	Gradient Ascent	0.094	0.415	0.573	0.945	0.709	0.469	0.978
	KGA	0.183	0.229	0.581	0.942	0.717	0.482	0.976
	KU	0.163	0.329	0.569	0.949	0.701	0.479	0.976
	UNLEARN ($\alpha = 0$)	0.118	0.263	0.567	0.966	0.702	0.466	0.976
	UNLEARN ($\alpha = 0.25$)	0.119	0.332	0.577	0.952	0.717	0.468	0.980
	UNLEARN ($\alpha = 0.5$)	0.124	0.427	0.579	0.947	0.709	0.482	0.988
	UNLEARN ($\alpha = 0.75$)	0.133	0.514	0.575	0.946	0.711	0.479	0.985
	UNLEARN ($\alpha = 1.0$)	0.135	0.628	0.581	0.969	0.723	0.460	0.989
NQ	Gradient Ascent	0.483	0.184	0.554	0.940	0.693	0.477	0.963
	KGA	0.501	0.243	0.557	0.946	0.697	0.479	0.989
	KU	0.416	0.113	0.558	0.926	0.712	0.468	0.973
	UNLEARN ($\alpha = 0$)	0.419	0.142	0.570	0.936	0.717	0.464	0.979
	UNLEARN ($\alpha = 0.25$)	0.487	0.141	0.571	0.950	0.714	0.481	0.984
	UNLEARN ($\alpha = 0.5$)	0.583	0.146	0.578	0.947	0.708	0.479	0.985
	UNLEARN ($\alpha = 0.75$)	0.655	0.146	0.569	0.941	0.711	0.474	0.991
	UNLEARN ($\alpha = 1.0$)	0.703	0.147	0.567	0.941	0.716	0.471	0.983

Table 5: Performance of UNLEARN when the rank (k) is modified. Targeted Task represents the task that was 'unlearned'. The tasks of interest are NarrativeQA (NQA), NaturalQuestions (NQ), Massive Multitask Language Understanding (MMLU), IMDB benchmark for sentiment analysis in movies (IMDB), Real-world Annotated Few-Shot (RAFT), Grade School Math 8K (GSM8K), and arithmetic. The green columns represent the targeted task and the yellow columns represent the similar task

k	Targeted Task	Evaluation Tasks						
		NQA	NQ	MMLU	IMDB	RAFT	GSM8K	arithmetic
Base Model		0.778	0.68	0.583	0.952	0.719	0.483	0.991
1	NQA	0.167	0.599	0.58	0.938	0.702	0.464	0.974
	NQ	0.684	0.198	0.582	0.951	0.701	0.482	0.989
2	NQA	0.151	0.609	0.564	0.931	0.712	0.479	0.987
	NQ	0.688	0.173	0.58	0.95	0.701	0.466	0.97
4	NQA	0.128	0.624	0.568	0.946	0.703	0.471	0.971
	NQ	0.711	0.152	0.567	0.934	0.718	0.482	0.986
8	NQA	0.136	0.627	0.58	0.931	0.718	0.475	0.99
	NQ	0.701	0.152	0.579	0.931	0.698	0.468	0.974
16	NQA	0.135	0.628	0.581	0.969	0.723	0.46	0.989
	NQ	0.703	0.147	0.567	0.941	0.716	0.471	0.983
32	NQA	0.134	0.619	0.579	0.937	0.704	0.467	0.974
	NQ	0.704	0.156	0.583	0.933	0.696	0.483	0.98