KNOWLEDGE EDITING WITH SUBSPACE-AWARE KEY-VALUE MAPPINGS

Haewon Park[1]*, Sangwoo Kim[2]*, Yohan Jo[1]†
[1]Graduate School of Data Science, Seoul National University
[2]Department of Linguistics, Seoul National University

{dellaanima2,hemy0101,yohan.jo}@snu.ac.kr

ABSTRACT

Knowledge editing aims to efficiently correct factual errors in Language Models (LMs). The popular locate-then-edit approach modifies an MLP layer by finding an optimal mapping between its input vector (key) and output vector (value) that leads to the expression of the edited knowledge. However, existing methods without any constraints on the key and value vectors cause significant perturbations to the edited model. To address this, we propose **Subspace Knowledge Edit (SUIT)**, a method that identifies and modifies only the subspace of critical features relevant to the edit. Our empirical results on `LLaMA-3-8B`, `GPT-J-6B`, and `Qwen2.5-7B` models show that SUIT dramatically improves knowledge preservation over strong baselines while maintaining high edit efficacy. This effectiveness confirms that SUIT successfully identifies the critical subspace for the edit. Further analyses provide additional validation for our approach. The source code and data will be released to the public upon publication of the paper.

1 INTRODUCTION

Large language models (LLMs) retain and recall substantial factual knowledge. However, they often produce incorrect statements due to noisy training data or a temporal shift (Maynez et al., 2020; Ji et al., 2023; Lin et al., 2022). These errors reveal gaps and temporal drift in the model's knowledge, indicating the need for edits to correct these issues. Fine-tuning is commonly used for this purpose, but it is computationally costly and susceptible to overfitting and catastrophic forgetting (Zhang et al., 2024; Bethune et al., 2025; Luo et al., 2025). Consequently, knowledge editing methods have emerged as a promising alternative, enabling targeted edits of specific knowledge while preserving the rest (Yao et al., 2023; Wang et al., 2024b). Among various knowledge editing methods (Wang et al., 2024a), our work builds on the *locate-then-edit* approach. This approach, which edits knowledge by directly identifying and updating the edit-relevant weights, has shown high precision in both mass and sequential editing (Meng et al., 2023b; Fang et al., 2025).

Knowledge editing replaces the old object o in a factual tuple (s, r, o) with a new object o^*. For example, for the subject $(s,$ *"Chrome"*$)$ and the relation $(r,$ *"was developed by"*$)$, the old object $(o,$ *"Google"*$)$ can be edited to the new object $(o^*,$ *"Apple"*$)$. Within the *locate-then-edit* methods, this editing is performed by viewing the Transformer MLP's down-projection matrix $\mathbf{W}$ as a linear associative memory (Anderson, 1972; Kohonen, 1972; Meng et al., 2023a), where $\mathbf{W}$ maps key vectors to value vectors. In these methods, the *key* vector $\mathbf{k}$ encodes the subject s, and the *value* vector $\mathbf{v}$ encodes the (r, o). The edit is then achieved by computing a new value vector $\mathbf{v}^*$ for the new pair (r, o^*) and redirecting the mapping from $\mathbf{k} \mapsto \mathbf{v}$ to $\mathbf{k} \mapsto \mathbf{v}^*$. Once the pair $(\mathbf{k}, \mathbf{v}^*)$ is specified, the corresponding weight update $\mathbf{\Delta}$ to be added to $\mathbf{W}$ such that $(\mathbf{W} + \mathbf{\Delta})\mathbf{k} = \mathbf{v}^*$ can be calculated in closed form, allowing the final weights to be set as $\mathbf{W}' = \mathbf{W} + \mathbf{\Delta}$. Thus, the result of the edit is determined by the specification of $\mathbf{k}$ and $\mathbf{v}^*$.

An ideal knowledge editing method should edit targeted knowledge while preserving unrelated knowledge. This latter property, known as specificity, prevents perturbation: any unintended change

*Equal contribution.
†Corresponding author.

in the model's output on unedited inputs. Understanding how the model represents knowledge is key to edit the knowledge without causing perturbation. According to the *Linear Representation Hypothesis*, a language model's hidden states are composed of a linear combination of semantic features, where each feature occupies a distinct subspace (Elhage et al., 2022; Mikolov et al., 2013). Prior research has empirically shown that these decomposable *feature*s encode interpretable information (Huang et al., 2024; Park et al., 2024). This perspective suggests that the key and value vectors themselves are also linear combinations of semantic *feature*s.

For knowledge editing, we hypothesize that these key and value vectors decompose into features relevant to the specific knowledge being edited and that are less edit-relevant. Therefore we propose **Subspace Knowledge Edit (SUIT)** that localizes target knowledge to specific subspaces and confines edits within them. For the key vector $\mathbf{k}$, we isolate its entity-specific subspaces containing features that activate differently for the various entities, removing identified common subspace. For the new value vector $\mathbf{v}^*$, we restrict the update within the subspace that primarily encodes the new object o^*.

Across `LLaMA-3-8B`, `GPT-J-6B`, and `Qwen2.5-7B`, **SUIT** demonstrates a significant leap in performance. Compared to AlphaEdit (Fang et al., 2025), a leading baseline renowned for minimizing knowledge disruption, SUIT achieves a substantial gain in specificity, improving by 43.2 points on `LLaMA-3-8B`, while retaining high edit efficacy. It also demonstrates robustness in preserving the model's general capabilities. To further validate our approach, we demonstrate **SUIT**'s effectiveness in reducing a entity's last token perturbation, empirically test our hypotheses on subspace identification, and present a hyperparameter analysis and an ablation study.

2 Related Work

Knowledge Editing Knowledge editing methods can be broadly categorized into three paradigms. *Memory-based approaches* preserve the original model by storing edited knowledge externally and retrieving it during inference (Mitchell et al., 2022; Hartvigsen et al., 2023). *Meta-learning approaches* employ auxiliary networks to learn weight updates that are subsequently applied to the base model for efficient editing (Mitchell et al., 2022; Cao et al., 2021). Finally, *Locate-Then-Edit approaches* identify knowledge-relevant parameters within the model and directly modify them to achieve targeted updates (Meng et al., 2023a;b).

Linear Representation Hypothesis The *Linear Representation Hypothesis* posits that the hidden states of language models are a linear combination of semantic features, with each feature occupying a distinct subspace (Elhage et al., 2022; Mikolov et al., 2013; Park et al., 2024). Indeed, numerous studies have empirically demonstrated this hypothesis, showing that a variety of features—such as syntax, position, and factual knowledge, among others—can be identified within specific, decomposable subspaces or directions (Huben et al., 2024; Ji et al., 2023; Huang et al., 2024).

3 Preliminaries

3.1 Linear Associative Memory

Knowledge editing in language models aims to edit a fact triplet from (s, r, o) to (s, r, o^*). The *locate-then-edit* methods (Meng et al., 2023a;b; Fang et al., 2025) achieve this by viewing the MLP's down-projection layer as a linear associative memory that maps keys to values. From this perspective, MLP's down-projection layer can be expressed as

$$\mathbf{W}\mathbf{k} = \mathbf{v}.$$

Here, the down-projection matrix $\mathbf{W}$ maps the key vector $\mathbf{k}$, which is the MLP's up-projection activation, to the value vector $\mathbf{v}$.

Based on this, the core idea of the *locate-then-edit* methods is that the key vector $\mathbf{k}$ encodes the subject s, while the value vector $\mathbf{v}$ encodes the relation r and object o. To edit the model's knowledge, these methods update the matrix $\mathbf{W}$ by adding an update matrix $\mathbf{\Delta}$. This change redirects the mapping of the key vector $\mathbf{k}$ from the value vector $\mathbf{v}$, encoding (r, o), to a new value vector $\mathbf{v}^*$

that encodes (r, o^*). This remapping is achieved by finding an update matrix $\mathbf{\Delta}$ that satisfies the approximation:

$$(\mathbf{W} + \mathbf{\Delta})\mathbf{k} \approx \mathbf{v}^*$$

The update matrix $\mathbf{\Delta}$ is calculated to satisfy the condition $\mathbf{\Delta k} \approx \mathbf{r}$, where $\mathbf{r} := \mathbf{v}^* - \mathbf{v}$, referred to as the residual vector.

Building on this principle, MEMIT (Meng et al., 2023b) extends the approach to edit multiple pieces of knowledge simultaneously. For a batch of n facts $(s, r, o^*)_i$ where $i = 1, \dots, n$, it first computes the corresponding key vectors $\mathbf{k}_i$ and residual vectors $\mathbf{r}_i$. These are then concatenated to form a key matrix $\mathbf{K} = [\mathbf{k}_1 \mid \mathbf{k}_2 \mid \cdots \mid \mathbf{k}_n]$ and a residual matrix $\mathbf{R} = [\mathbf{r}_1 \mid \mathbf{r}_2 \mid \cdots \mid \mathbf{r}_n]$. MEMIT provides a closed-form solution to find the update matrix $\mathbf{\Delta}$ using these matrices.

Subsequently, AlphaEdit (Fang et al., 2025), based on MEMIT, introduced a new formula for $\mathbf{\Delta}$ designed to better preserve the model's existing knowledge. The proposed formula is:

$$\mathbf{\Delta} = \mathbf{R}\mathbf{K}^\top\mathbf{P}\left(\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P} + \mathbf{K}\mathbf{K}^\top\mathbf{P} + \mathbf{I}\right)^{-1} \tag{1}$$

In this formula, both $\mathbf{K}_p$ and $\mathbf{P}$ are pre-computed (details in Appendix A). In other words, since the computation of the update matrix $\mathbf{\Delta}$ depends on the **key vector** $\mathbf{k}$ for the subject s and the **residual vector** $\mathbf{r}$ representing the change from o to o^*, it is crucial to calculate these values accurately.

3.2 Computing the Key Vector $\mathbf{k}$ and the residual Vector δ

Key Vector. Suppose we want to edit the fact triplet from (s, r, o) to (s, r, o^*). To compute the key vector $\mathbf{k}$, we extract the MLP up-projection activation from the last token of the subject (e.g., *"rome"* in *"Chrome"*). To improve the generalization of the key vector, we repeat this with various prefixes. We average the extracted MLP up-projection activations to obtain the final key vector $\mathbf{k}$.

Residual Vector. While ROME (Meng et al., 2023a) targeted a single layer, most subsequent approaches perform edits across multiple layers. These multi-layer methods do not compute the residual vector $\mathbf{r}$ separately for each layer. Instead, they first calculate an *entire residual vector* δ (henceforth, the residual vector) from the residual stream of the final layer being edited. This vector δ is then distributed proportionally to determine the specific $\mathbf{r}$ for each layer involved in the edit. To obtain δ, it is added to the residual stream $\mathbf{h}$ at the subject's last token position in the last modified layer and optimized via gradient descent to maximize the logit of the new object o^* (*"Apple"*) given the input s, r (*"Chrome was developed by"*). To prevent overfitting to the new fact, which can lead to the corruption of unrelated knowledge, a regularization term $\mathcal{R}$ is added to the loss function (details in Appendix B). The optimization objective is formulated as:

$$\delta = \arg\min_{\tilde{\delta}} \left\{ -\log p\Big(o^* \mid \mathbf{h}^* \leftarrow \mathbf{h} + \tilde{\delta}\Big) + \mathcal{R} \right\}. \tag{2}$$

4 SUIT: Subspace Knowledge Edit

4.1 Knowledge Editing under Linear Representation Hypothesis

According to the *Linear Representation Hypothesis*, the key and value vectors within an MLP's down-projection layer can be viewed as a composition of semantic features. For knowledge editing, we hypothesize that these vectors consist of features that are either relevant or irrelevant to the specific edit. The key vector $\mathbf{k}$, which encodes the subject, can be divided into entity-specific features and more general, entity-agnostic features that activate similarly across many subjects. Similarly, the value vector $\mathbf{v}$, encoding the relation and object, contains features that primarily define the object (o or o^*) alongside other less relevant features.

To ensure that knowledge edits are precise—modifying only the target knowledge while preserving other knowledge—we propose introducing explicit constraints. When computing the key vector $\mathbf{k}$, we aim to consider only the subspace occupied by its entity-specific features. Likewise, when computing the residual vector $\boldsymbol{\delta}$, we aim to consider only the feature directions that significantly influence the object's logit. Accordingly, in the following sections we obtain the subspace-aware key vector $\mathbf{k}'$ (§ 4.2) and the subspace-aware residual vector $\boldsymbol{\delta}'$ (§ 4.3); using these, we compute the update matrix $\mathbf{\Delta}$ via Eq. (1).

4.2 Subspace-Aware Computation for Our Key Vector $\mathbf{k}'$

When computing our key vector $\mathbf{k}'$ that encodes the subject s, our objective is to isolate the component that lies within $\mathcal{K}_s$—the subspace containing only entity-specific features, which activate differently for the various entities. Conversely, the orthogonal subspace $\mathcal{K}_s^{\perp}$ contains the entity-agnostic features that activate similarly across many subjects.

We begin with the key vector $\mathbf{k}$. (§ 3.2) We then decompose this vector into its entity-specific component, $\mathbf{k}_s \in \mathcal{K}_s$, and its entity-agnostic component, $\mathbf{k}_{\sim s} \in \mathcal{K}_s^{\perp}$. Our key vector $\mathbf{k}'$ is obtained by removing this entity-agnostic component:

$$\mathbf{k} = \mathbf{k}_s + \mathbf{k}_{\sim s}$$
$$\mathbf{k}' = \mathbf{k} - \mathbf{k}_{\sim s} = \mathbf{k}_s$$

The core task is to identify the entity-agnostic subspace $\mathcal{K}_s^{\perp}$ and compute these vector components. We accomplish this through the following procedure.

We sample $N = 10{,}000$ subjects from ParaRel(Elazar et al., 2021), a dataset of (s, r, o) triplets derived from Wikidata. For each subject, we compute its key vector $\mathbf{k}$ to form the matrix: $\mathbf{K}_{\text{subject}} = [\mathbf{k}_1 \mid \mathbf{k}_2 \mid \cdots \mid \mathbf{k}_{10000}]$.

Applying singular value decomposition (SVD) to this matrix yields:

$$\mathbf{K}_{\text{subject}} = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^{\top},$$

where $\mathbf{S} = \operatorname{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r)$ with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ denotes the singular values, and $\mathbf{U} = [\mathbf{u}_1 \mid \mathbf{u}_2 \mid \cdots \mid \mathbf{u}_r]$ contains the corresponding left singular vectors.

To determine how many critical components to remove, we introduce a hyperparameter $\tau_{\text{energy}} \in [0, 1)$ that represents the proportion of total variance (energy) to isolate. Let $E_{\text{total}} = \sum_{i=1}^{r} \sigma_i^2$ be the total energy of $\mathbf{K}_{\text{subject}}$. We find the smallest integer m such that the cumulative energy of the first m components exceeds this threshold: $\sum_{i=1}^{m} \sigma_i^2 \;\geq\; \tau_{\text{energy}} \cdot E_{\text{total}}$. Let $\mathbf{U}_{\text{t}} = [\mathbf{u}_1 \mid \cdots \mid \mathbf{u}_m]$ denote the matrix of the first m left singular vectors. We then define the entity-agnostic subspace $\mathcal{K}_s^{\perp}$ as their span:

$$\mathcal{K}_s^{\perp} := \operatorname{span}(\mathbf{U}_{\text{t}})\,.$$

This subspace $\mathcal{K}_s^{\perp}$ represents the directions of common features that activate similarly across various entities. Finally, we use the matrix $\mathbf{U}_{\text{t}}\mathbf{U}_{\text{t}}^{\top}$ to project the key vector $\mathbf{k}$ onto the entity-agnostic subspace $\mathcal{K}_s^{\perp}$. By subtracting this projection $\mathbf{k}_{\sim s}$, we remove the entity-agnostic component, leaving only the entity-specific component $\mathbf{k}_s$. This procedure yields the constrained key vector $\mathbf{k}'$, which now contains only the *features* relevant to the subject being edited:

$$\mathbf{k}' = \mathbf{k} - \mathbf{k}_{\sim s}, \quad \mathbf{k}_{\sim s} = \mathbf{U}_{\text{t}}\mathbf{U}_{\text{t}}^{\top}\mathbf{k}.$$

4.3 Subspace-Aware Computation for Our Residual Vector δ'

The computation of the residual vector δ is intended to modify the residual stream $\mathbf{h}$ to encode (r, o^*). (§ 3.2) Rather than altering the full-dimensional residual stream $\mathbf{h}$, our approach is to target a low-dimensional subspace that governs the model's prediction for the given (s, r) pair.

We hypothesize that this targeted modification can be achieved within a two-dimensional subspace spanned by two critical unit feature directions, $\mathbf{w}_1$ and $\mathbf{w}_2$. Specifically, increasing the magnitude of $\mathbf{h}$ along $\mathbf{w}_1$ (i.e., $\mathbf{h}^{\top}\mathbf{w}_1$) raises the logit of the new object o^*, while decreasing its magnitude along $\mathbf{w}_2$ (i.e., $\mathbf{h}^{\top}\mathbf{w}_2$) suppresses the logit of the old object o. To edit o to o^*, we swap these magnitudes: we increase the magnitude of $\mathbf{h}$ along $\mathbf{w}_1$ and decrease its magnitude along $\mathbf{w}_2$. For simplicity, we ignore interactions between the two directions and implement the edit as a simple additive update (details in Appendix C). The updated residual stream $\mathbf{h}^*$ is:

$$\mathbf{h}^* = \mathbf{h} + \delta', \quad \delta' = (\mathbf{h}^{\top}\mathbf{w}_2 - \mathbf{h}^{\top}\mathbf{w}_1)\mathbf{w}_1 + (\mathbf{h}^{\top}\mathbf{w}_1 - \mathbf{h}^{\top}\mathbf{w}_2)\mathbf{w}_2$$

The process for finding the optimal basis vectors, $\{\mathbf{w}_1, \mathbf{w}_2\}$, follows a similar structure to the optimization for the residual vector δ shown in Eq. (2). The primary objective is to maximize the

	Method	Counterfact							zsRE			
		S	Eff. ↑	Gen. ↑	Spe. ↑	Flu. ↑	Con. ↑	GC ↑	S	Eff. ↑	Gen. ↑	Spe. ↑
LLama3	Pre-edit	0.0	0.0	0.0	100.0	634.9	20.9	63.4	45.1	35.9	34.8	100.0
	FT-L	1.9	9.9	1.4	1.3	438.5	19.2	6.2	43.2	34.8	34.0	88.0
	MEND	0.0	0.0	0.0	1.3	519.5	0.6	0.0	0.0	0.0	0.0	**89.7**
	MEMIT	48.3	76.2	74.0	28.2	628.9	36.7	60.8	68.3	89.3	85.7	47.5
	PMET	37.6	56.6	56.0	22.6	608.8	33.8	50.1	69.8	89.2	83.9	50.4
	AlphaEdit	55.8	97.3	88.7	31.0	**633.6**	**38.6**	62.2	73.7	93.5	**88.7**	53.3
	SUIT	**86.8**	**99.7**	**90.3**	**74.2**	631.2	38.2	**63.0**	**81.6**	**95.2**	85.7	68.5
GPT-J	Pre-edit	0.0	0.0	0.0	100.0	621.1	23.9	24.3	35.4	27.2	26.3	100.0
	FT-L	13.3	64.9	46.7	5.3	334.2	12.2	**24.2**	28.4	69.3	60.6	13.4
	MEND	0.0	0.0	0.0	0.0	515.0	2.7	0.0	0.6	0.4	0.4	80.0
	MEMIT	60.2	92.0	90.4	35.8	617.4	48.5	20.1	88.6	96.8	90.8	79.9
	PMET	57.4	84.6	84.5	35.0	618.5	44.8	19.1	85.9	95.1	89.2	75.8
	AlphaEdit	73.0	98.3	**95.0**	49.0	**621.8**	**49.9**	19.5	**96.9**	99.1	**92.4**	**99.4**
	SUIT	**82.3**	**98.6**	93.3	**64.1**	619.4	49.4	20.4	95.9	**99.7**	89.5	99.3
Qwen2.5	Pre-edit	0.0	0.0	0.0	100.0	625.5	21.9	28.9	29.5	22.5	21.2	100.0
	FT-L	10.3	47.9	31.7	4.2	476.7	4.5	0.0	7.3	18.1	15.6	3.4
	MEND	0.0	0.0	0.0	0.0	466.3	0.1	0.0	0.0	0.0	0.0	85.3
	MEMIT	22.6	83.0	84.0	9.2	622.1	37.0	**34.8**	72.7	82.9	72.2	65.1
	PMET	32.6	67.5	65.9	16.1	545.2	27.5	14.8	60.2	71.9	65.5	48.4
	AlphaEdit	67.8	97.1	**91.6**	43.4	**626.2**	**41.0**	28.1	**89.6**	**97.5**	**86.3**	85.8
	SUIT	**85.7**	**99.5**	86.8	**74.4**	**626.2**	37.4	30.8	88.2	93.9	76.9	**96.6**

Table 1: Results on COUNTERFACT and ZSRE. Best numbers are **bold**; second-best are underlined. Abbreviations: Eff. = Efficacy, Gen. = Generalization, Spe. = Specificity, Flu. = Fluency, Con. = Consistency, GC = General Capability.

logit of the new object o^*. While Eq. (2) included a general regularization term $\mathcal{R}$, it is unnecessary in our approach as we constrain the update to a two-dimensional subspace. To encourage the basis vectors $\mathbf{w}_1$ and $\mathbf{w}_2$ to represent distinct directions, we introduce a directional penalty term, formulated as $\left(\hat{\mathbf{w}}_1^\top \hat{\mathbf{w}}_2\right)^2$. Here, since $\{\mathbf{w}_1, \mathbf{w}_2\}$ are completely symmetric in the formulation, we can assume the constraint $\mathbf{h}^\top \mathbf{w}_1 < \mathbf{h}^\top \mathbf{w}_2$ without loss of generality. The complete optimization objective reflecting this is therefore formulated as:

$$\{\mathbf{w}_1, \mathbf{w}_2\} = \arg\min_{\hat{\mathbf{w}}_1, \hat{\mathbf{w}}_2} \left\{ -\log p\left(o^* \mid \mathbf{h}^* \leftarrow \mathbf{h} + \hat{\delta}'\right) + \lambda \left(\hat{\mathbf{w}}_1^\top \hat{\mathbf{w}}_2\right)^2 \right\},$$

where the hyperparameter λ is the penalty weight that controls the strength of the directional penalty.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Models & Baseline Methods. We conduct our experiments on these models: Llama3-Instruct (8B) (Grattafiori et al., 2024), GPT-J (6B) (Wang & Komatsuzaki, 2021), and Qwen2.5-Instruct (7B) (Yang et al., 2024). We compare SUIT against several representative model editing baselines, including Fine-Tuning (FT-L) (Zhu et al., 2020), MEND (Mitchell et al., 2022), PMET (Li et al., 2024), MEMIT (Meng et al., 2023b), and AlphaEdit (Fang et al., 2025). Further comparison results with other methods (FT-W (Zhu et al., 2020), ROME (Meng et al., 2023a), RECT (Gu et al., 2024), PRUNE (Ma et al., 2024), NSE (Jiang et al., 2024)) are presented in Appendix F.1, and the hyperparameters are listed in Appendix D.2.

Dataset and Metrics We evaluate model editing performance on two widely used benchmarks: COUNTERFACT(Meng et al., 2023a) and ZSRE(Levy et al., 2017). In line with prior works (Meng et al., 2023a;b; Fang et al., 2025), we employ three metrics to evaluate the performance of edit: *Efficacy*, *Generalization*, and *Specificity*. First, *Efficacy* assesses whether the model generates o^* for a given *rewrite prompt* (s, r), while *Generalization* measures the same for *paraphrase prompts*.

Lastly, *Specificity* verifies that an edit does not negatively impact unrelated knowledge. This is evaluated using a *neighborhood prompt*, which involves a different subject s' but shares the same r and o as the fact being edited. To provide a single, comprehensive measure, we also report the harmonic mean of these three metrics, denoted as S. Beyond factual correctness, we also evaluate generation quality via *Consistency* and *Fluency*. Further details are provided in the Appendix E.

Evaluation Details Our objective is *editing*, the task of changing (s, r, o) to (s, r, o^*). Since editing presupposes the model's prior knowledge of a fact, we exclusively evaluate on instances where the model's argmax prediction is o for the original prompt (s, r) as well as its corresponding paraphrase and neighborhood prompts. This condition is not applied to ZSRE, which contrasts with declarative datasets like COUNTERFACT. Its question-based prompts lead the model to predict sentence-starters (e.g., "The") rather than the immediate object, making the argmax check unsuitable. In Table 1, we present the results of applying 1000 edits sequentially in 10 batches of 100. For this evaluation, an edit is considered successful based on a *generation-based criterion*, which counts an edit as successful only if o^* is the argmax prediction. In the Appendix F.2, we also report results with the less demanding *probability-based criterion* ($P(o^*) > P(o)$), common in prior work.

General Capability We evaluate the model's General Capability (GC) to measure any side effects from editing. The GC score (Table 1) is the average F1 score across six benchmarks: MMLU (Hendrycks et al., 2021b) and tasks from GLUE (Wang et al., 2019) (NLI, MRPC, SST, RTE, CoLA). Detailed results for each benchmark can be found in the Appendix F.3.

5.2 EXPERIMENTAL RESULTS

As shown in Table 1, SUIT attains the highest overall performance (*S* score) across most model–dataset pairs. Its advantage is most evident on COUNTERFACT: with Llama, SUIT achieves an *S* score of 86.8%, 31 points lead over AlphaEdit (55.8%). The gain is driven by *Specificity*, where SUIT scores 74.2% compared to 31.0% (+43.2 points). The improvement extends to GPT-J (+15.1 points) and Qwen (+31.0 points).We further assessed contextual robustness by evaluating the model edited with COUNTERFACT on the CHED (Park et al., 2025) dataset, and Appendix F.4 confirms that SUIT remains effective even with preceding context. On the ZSRE dataset, SUIT's strong performance continues. It achieves the highest *S* score on Llama (81.6%). In addition, it records the best Efficacy on GPT-J (99.7%) and the highest Specificity on Qwen (96.6%).

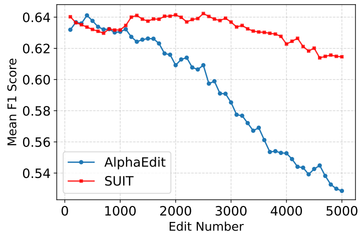
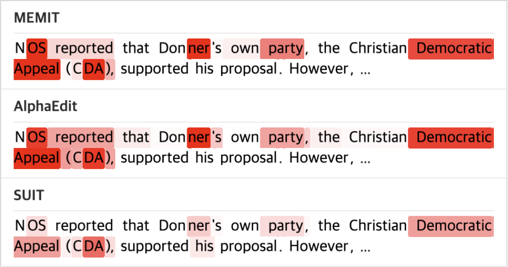


Figure 1: Mean F1 score degradation during a 5,000-edit setting.

Beyond these metrics, SUIT preserves Fluency, Consistency, and overall General Capability, showing that it edits knowledge without degrading overall performance. We further validate this in a 5,000-edit setting, tracking General Capability degradation at every 100-edit interval (Figure 1). As shown in the figure, SUIT remains highly stable and surpasses AlphaEdit while maintaining editing efficacy. Complete results and benchmark-level F1 scores are reported in Appendix F.5. In summary, SUIT enables effective knowledge editing with minimal disruption, setting a new standard for scalable and reliable model editing.

6 ANALYSIS

To further investigate SUIT, we conducted a series of analyses. We demonstrate its effectiveness in reducing perturbation at the subject's last-token. We then empirically validate our hypotheses regarding the subspaces identified for computing the key and residual vectors. Finally, we present a hyperparameter tradeoff analysis and an ablation study. All analyses were performed using the Llama-3-Instruct model. For experiments requiring edited models, we used models edited with 10 batches of 100 edits each from the COUNTERFACT.



MEMIT

NOS reported that Donner's own party, the Christian Democratic Appeal (CDA), supported his proposal. However, ...

AlphaEdit

NOS reported that Donner's own party, the Christian Democratic Appeal (CDA), supported his proposal. However, ...

SUIT

NOS reported that Donner's own party, the Christian Democratic Appeal (CDA), supported his proposal. However, ...

Figure 2: Comparison of token-level perturbations in residual streams.

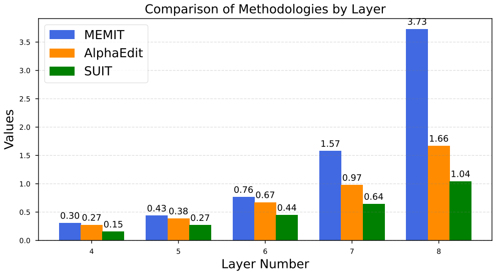


Figure 3: Norm differences of MLP outputs at the last token position across methods.

6.1 Reducing Entity's Last Token Perturbation

Ideally, an edited model is expected to behave identically to the original model on unedited knowledge, without introducing any perturbation. Prior research (Meng et al., 2023a; Geva et al., 2023; Chughtai et al., 2024) has demonstrated that an entity's attributes are predominantly enriched at the last token position of the entity. Accordingly, *locate-then-edit* methods perform edits by deriving key and value vectors directly from this position. This approach, however, induces significant perturbation precisely at this critical location.

Figure 2 shows the L_2 norm of difference in the residual streams of the final edited layer between the original model and models edited using MEMIT, AlphaEdit, and SUIT. This difference, representing the perturbation at each token, is visualized through color intensity on a sample paragraph from the Wikinews Article Dataset. The figure demonstrates that, compared to other tokens, all methods generally exhibit a more notable perturbation at the last token of the subject entity (e.g., *"OS"* in *"NOS"*, *"ner"* in *"Donner"*). Notably, among these methods, the perturbation from SUIT is visibly less pronounced than that of MEMIT and AlphaEdit. Further examples confirming this pattern are provided in the Appendix H.

To quantify the perturbation from editing, we analyze the output of the MLP layers, which are the direct targets of the edits. Using unedited knowledge from the COUNTERFACT dataset, we measure the L_2 distance between the outputs of the original and edited models at the subject entity's last token position. As shown in Figure 3, while all three methods induce a non-zero perturbation, SUIT consistently exhibits the smallest change. This indicates that SUIT performs more localized and precise edits, affecting unrelated knowledge the least.

6.2 Analysis for subspaces

In this section, we investigate whether the subspaces $\mathcal{K}_s$, $\mathcal{K}_s^{\perp}$ and the directions $\mathbf{w}_1$, $\mathbf{w}_2$, which we identified to find the key vector $\mathbf{k}$ and the residual vector δ, indeed correspond to the feature subspaces we hypothesized. We hypothesized that $\mathcal{K}_s$ is a entity-specific feature subspace that activates differently for each subject entity. Conversely, its orthogonal subspace $\mathcal{K}_s^{\perp}$ is assumed to be a entity-agnostic feature space, activating similarly across different entities. Furthermore, we posited that $\mathbf{w}_1$, $\mathbf{w}_2$ are crucial directions for the update. Specifically, when their scaled versions are added to the residual stream $\mathbf{h}$, we believe $\mathbf{w}_1$ is the principal direction for increasing the logit of the new object o^*, while $\mathbf{w}_2$ is principal direction for decreasing the logit of the old object o.

6.2.1 Analysis for $\mathcal{K}_s$, $\mathcal{K}_s^{\perp}$

To investigate whether $\mathcal{K}_s$ is entity-specific and $\mathcal{K}_s^{\perp}$ is entity-agnostic, we designed an experiment to measure component variance. We began by extracting the key vectors $\mathbf{k}$ for 5,000 randomly selected subjects from the COUNTERFACT and ZSRE datasets. Each key vector $\mathbf{k}$ was then decomposed into its component $\mathbf{k}_s$ in the subspace $\mathcal{K}_s$ and its component $\mathbf{k}_{\sim s}$ in the orthogonal space $\mathcal{K}_s^{\perp}$. Finally, we computed the variance for each set of components across all 5,000 subjects.

The results of our variance analysis, presented in Table 2, align with our initial expectations. Across both datasets, the variance of the entity-specific components $\mathbf{k}_s$ was significantly higher than that

Table 2: Variance of decomposed key vector components across 5,000 subjects.

	COUNTERFACT	ZSRE
$V(\mathbf{k}_{\sim s})$	2.041	1.333
$V(\mathbf{k}_s)$	5.269	5.938

Table 3: Proportion of the modification affecting entity-agnostic components at layer 4.

Prompt type	MEMIT	AlphaEdit	SUIT
rewrite	0.2814	0.4623	0.0035
paraphrase	0.2929	0.4717	0.0039
neighborhood	0.6805	0.8114	0.0201

of the entity-agnostic ones $\mathbf{k}_{\sim s}$, being approximately 2.6 times higher for COUNTERFACT and 4.5 times higher for ZSRE. This finding is consistent with our hypothesis, suggesting that $\mathcal{K}_s$ may capture entity-specific features that vary across individuals, while $\mathcal{K}_s^{\perp}$ appears to contain more stable, entity-agnostic information.

Furthermore, we conducted an experiment to verify that our update matrix, $\mathbf{\Delta}$, interacts less with the subspace $\mathcal{K}_s^{\perp}$. To test this, we compared our method against MEMIT and AlphaEdit on the rewrite, paraphrase, and neighborhood prompts in the COUNTERFACT. Specifically, we measured the proportion of the update that affects the entity-agnostic components, calculated as $\|\Delta\mathbf{k}_{\sim s}\|^2/\|\Delta\mathbf{k}\|^2$. The results for layer 4, the first edited layer, are presented in Table 3, while the results for the other layers can be found in the Appendix.

The results clearly demonstrate that for all prompt types, the proportion of the update affecting the entity-agnostic space $\|\Delta\mathbf{k}_{\sim s}\|^2/\|\Delta\mathbf{k}\|^2$ is negligible for our method, SUIT. This is in stark contrast to MEMIT and AlphaEdit, which show a significantly higher proportion of their modifications impacting these common components. This finding suggests that SUIT successfully isolates its edits to the entity-specific space $\mathcal{K}_s$, leaving the shared, entity-agnostic knowledge largely untouched. We can, therefore, infer that this precise targeting is a key reason for SUIT's enhanced specificity for neighborhood prompts.

6.2.2 ANALYSIS FOR $\mathbf{w}_1$, $\mathbf{w}_2$

Next, we verified our hypothesis that the subspace spanned by $\mathbf{w}_1$ and $\mathbf{w}_2$ represents the critical directions for increasing the logit of the new object o^*. To do this, we first computed the residual vector δ. We then decomposed this vector into two distinct components: the component lying in the subspace $\mathrm{span}(\mathbf{w}_1, \mathbf{w}_2)$, $\delta_{\|W}$, and the remaining orthogonal component, $\delta_{\perp W}$. The projection is calculated as

$$\delta_{\|W} = P_W\delta, \quad \text{where } W = [\mathbf{w}_1,\ \mathbf{w}_2], \quad P_W = W\big(W^TW\big)^{-1}W^T.$$

Table 4: Results of steering with decomposed components of δ.

Space	$\delta_{\|W}$	$\delta_{\perp W}$
$\|\delta_{\|/\perp W}\|^2/\|\delta\|^2$	24.17	75.82
$p(o^*)$	0.67	0.59
logit(o^*)	-1.44	-1.72

To compare the respective effects of these two components on increasing the logit of o^*, we added each of $\delta_{\|W}$ and $\delta_{\perp W}$ to the residual stream $\mathbf{h}$ and measured the logit and probability of o^*, given the subject s and relation r, averaged over the 1000 edits in COUNTERFACT.

The results, presented in Table 4, reveal a compelling finding. $\delta_{\|W}$ accounts for only about 24% of the total squared norm of δ. Despite its significantly smaller magnitude, this component was more effective at increasing the logit and probability of o^* than the remaining 76% of the vector $\delta_{\perp W}$. This outcome indicates that while the residual vector δ contains both components crucial for increasing the logit of o^* and other less-essential ones, our approach computes our residual vector exclusively from the critical subspace, $\mathrm{span}(\mathbf{w}_1, \mathbf{w}_2)$. This allows our method to perform a more focused and potent update within a much narrower directional space.

We further analyzed the individual roles of $\mathbf{w}_1$ and $\mathbf{w}_2$ to test if $\mathbf{w}_1$ primarily increases the logit of the new object o^* while $\mathbf{w}_2$ primarily decreases the logit of the old object o. To do this, we decomposed our update vector δ' into its components along these directions, $\Delta\mathbf{w}_1 = (\mathbf{h}^\top\mathbf{w}_2 - \mathbf{h}^\top\mathbf{w}_1)\mathbf{w}_1$, $\Delta\mathbf{w}_2 = (\mathbf{h}^\top\mathbf{w}_1 - \mathbf{h}^\top\mathbf{w}_2)\mathbf{w}_2$. We then incrementally added each component to the residual stream $\mathbf{h}$ by scaling it with an interpolation factor $k \in [0, 1]$ and observed the logits for both objects.

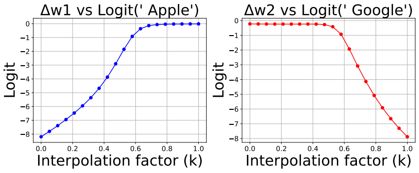


Figure 4: Effects of $\Delta\mathbf{w}_1$ and $\Delta\mathbf{w}_2$ on the logits of “Apple” and “Google”

Figure 4 shows the results for the edit (*“Chrome”*, *“was developed by”*, *“Apple”*). As expected, the $\Delta\mathbf{w}_1$ component is effective at increasing the target o^* (*“Apple”*) logit, while the $\Delta\mathbf{w}_2$ component effectively decreases the original o (*“Google”*) logit.

Although $\mathbf{w}_1$ and $\mathbf{w}_2$ drive logits as expected and were trained in different directions, we find that $\mathbf{w}_1$ also suppresses the old object o, and $\mathbf{w}_2$ promotes the new object o^*, rather than each playing only a single role. The effectiveness of fully disentangling them would be worth exploring in future work. For a detailed visualization and a full breakdown of these component effects, please see Appendix G.

6.3 Tradeoff and Ablation Study

In this section, we analyze the performance tradeoff between our method's two main components—key vector k' computation (§ 4.2) and residual vector δ' computation (§ 4.3)—by varying their respective hyperparameters, the energy threshold τ_{energy} and the penalty weight λ. We also conduct an ablation study to validate the contributions of each component.

Hyperparameter Tradeoff. For the analysis, we incrementally varied each hyperparameter from 0 to 0.9 while keeping the other fixed ($\lambda = 0.3$ or $\tau_{\text{energy}} = 0.4$), evaluating performance over 10 batches of 100 edits on the CounterFact. Regarding the energy threshold τ_{energy} , the efficacy score tended to remain stable as the threshold increased. The generation score showed a sweet spot around 0.3-0.4, which appears to be effective for identifying a suitable subspace $\mathcal{K}_s^{\perp}$. As τ_{energy} increased further, the subspace considered for editing tended to narrow. This created a clear tradeoff: the generation score decreased as the model became less fitted to the edits, while the specificity score increased.

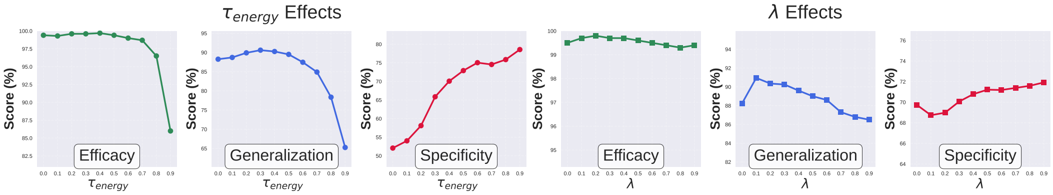


Figure 5: Tradeoff analysis for hyperparameters τ_{energy} and λ.

For the penalty weight λ, efficacy also showed a tendency to remain consistent. A higher λ tended to impose stronger regularization, compelling weight vectors $\mathbf{w}_1$ and $\mathbf{w}_2$ to find more divergent directions. This generally resulted in the model being less fitted to the edits, revealing a direct tradeoff where the generation score decreased while the specificity score increased.

Ablation Study. To isolate the contribution of each of our proposed components, we conducted an ablation study on the key vector computation (**k**-only) and the residual vector computation (δ-only), with the results summarized in Table 5. The study reveals a powerful synergy between the two components. Our full method not only preserves the high efficacy of the δ-only method and the strong specificity of the **k**-only method but also achieves a generalization score that significantly surpasses both. This result validates the effectiveness of our integrated approach.

Table 5: Performance analysis of the individual components of SUIT.

Method	Eff.	Gen.	Spe.
SUIT ($k + \delta$)	99.7	90.3	74.2
k-only	96.4	77.9	74.7
δ-only	99.7	83.8	44.6

7 Conclusion

In this work, we proposed **Subspace Knowledge Edit (SUIT)**, a method for targeted knowledge editing in language models. Grounded in the Linear Representation Hypothesis, SUIT constrains edits to edit-relevant subspaces by decomposing the key vector $\mathbf{k}$ into entity-specific feaures and restricting the residual vector δ to features relevant to the new object. This subspace-aware formulation enables precise modification of target knowledge. Our experiments demonstrate that SUIT achieves high efficacy and markedly improves specificity, confirming that subspace-based editing enables accurate knowledge updates while preserving model's general capabilities.

References

James A. Anderson. A simple neural network generating an interactive memory. *Mathematical Biosciences*, 14(3):197–220, 1972. ISSN 0025-5564. doi: https://doi.org/10.1016/0025-5564(72)90075-2. URL https://www.sciencedirect.com/science/article/pii/0025556472900752.

Luisa Bentivogli, Ido Dagan, Myroslava Dzikovska, Danilo Giampiccolo, and Bernardo Magnini. The fifth pascal recognizing textual entailment challenge. In *Proceedings of the Text Analysis Conference (TAC)*, volume 9, 2009.

Louis Bethune, David Grangier, Dan Busbridge, Eleonora Gualdoni, Marco Cuturi, and Pierre Ablin. Scaling laws for forgetting during finetuning with pretraining data injection, 2025. URL https://arxiv.org/abs/2502.06042.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models, 2021. URL https://arxiv.org/abs/2104.08164.

Bilal Chughtai, Alan Cooney, and Neel Nanda. Summing up the facts: Additive mechanisms behind factual recall in LLMs, 2024. URL https://openreview.net/forum?id=P2gnDEHGu3.

Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031, 2021. doi: 10.1162/tacl_a_00410. URL https://aclanthology.org/2021.tacl-1.60/.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition, 2022. URL https://arxiv.org/abs/2209.10652.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained model editing for language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HvSytvg3Jh.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12216–12235, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.751. URL https://aclanthology.org/2023.emnlp-main.751/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, and Abhishek Kadian et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Jia-Chen Gu et al. Model editing harms general abilities of large language models: Regularization to the rescue. *arXiv preprint arXiv:2401.04700*, 2024. URL https://arXiv.org/abs/2401.04700.

Thomas Hartvigsen, Swami Sankaranarayanan, Hamid Palangi, Yoon Kim, and Marzyeh Ghassemi. Aging with grace: Lifelong model editing with discrete key-value adaptors, 2023. URL https://arxiv.org/abs/2211.11031.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2021b. URL https://arxiv.org/abs/2009.03300.

Jing Huang, Zhengxuan Wu, Christopher Potts, Mor Geva, and Atticus Geiger. RAVEL: Evaluating interpretability methods on disentangling language model representations. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8669–8687, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.470. URL https://aclanthology.org/2024.acl-long.470/.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12), March 2023. ISSN 0360-0300. doi: 10.1145/3571730. URL https://doi.org/10.1145/3571730.

Houcheng Jiang, Junfeng Fang, Tianyu Zhang, An Zhang, Ruipeng Wang, Tao Liang, and Xiang Wang. Neuron-level sequential editing for large language models. *arXiv preprint arXiv:2410.04045*, 2024. URL https://arXiv.org/abs/2410.04045.

Teuvo Kohonen. Correlation matrix memories. *IEEE Transactions on Computers*, C-21(4):353–359, 1972. doi: 10.1109/TC.1972.5008975.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In Roger Levy and Lucia Specia (eds.), *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 333–342, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-1034. URL https://aclanthology.org/K17-1034/.

Xiaopeng Li, Shasha Li, Shezheng Song, Jing Yang, Jun Ma, and Jie Yu. Pmet: Precise model editing in a transformer, 2024. URL https://arxiv.org/abs/2308.08742.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.229. URL https://aclanthology.org/2022.acl-long.229/.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning, 2025. URL https://arxiv.org/abs/2308.08747.

Jun-Yu Ma, Hong Wang, Hao-Xiang Xu, Zhen-Hua Ling, and Jia-Chen Gu. Perturbation-restrained sequential model editing. *arXiv preprint arXiv:2405.16821*, 2024. URL https://arXiv.org/abs/2405.16821.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. On faithfulness and factuality in abstractive summarization. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1906–1919, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.173. URL https://aclanthology.org/2020.acl-main.173/.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt, 2023a. URL https://arxiv.org/abs/2202.05262.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. Mass editing memory in a transformer. *The Eleventh International Conference on Learning Representations (ICLR)*, 2023b.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013. URL https://arxiv.org/abs/1301.3781.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale, 2022. URL https://arxiv.org/abs/2110.11309.

Haewon Park, Gyubin Choi, Minjun Kim, and Yohan Jo. Context-robust knowledge editing for language models. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 10360–10385, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.540. URL https://aclanthology.org/2025.findings-acl.540/.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models, 2024. URL https://arxiv.org/abs/2311.03658.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, 2018.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019. URL https://arxiv.org/abs/1804.07461.

Ben Wang and Aran Komatsuzaki. Gpt-j-6b: A 6 billion parameter autoregressive language model, 2021.

Peng Wang, Ningyu Zhang, Bozhong Tian, Zekun Xi, Yunzhi Yao, Ziwen Xu, Mengru Wang, Shengyu Mao, Xiaohan Wang, Siyuan Cheng, Kangwei Liu, Yuansheng Ni, Guozhou Zheng, and Huajun Chen. EasyEdit: An easy-to-use knowledge editing framework for large language models. In Yixin Cao, Yang Feng, and Deyi Xiong (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 82–93, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-demos.9. URL https://aclanthology.org/2024.acl-demos.9/.

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, and Jundong Li. Knowledge editing for large language models: A survey, 2024b. URL https://arxiv.org/abs/2310.16218.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122, 2018.

An Yang et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arXiv.org/abs/2412.15115.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10222–10240, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.632. URL https://aclanthology.org/2023.emnlp-main.632/.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, Siyuan Cheng, Ziwen Xu, Xin Xu, Jia-Chen Gu, Yong Jiang, Pengjun Xie, Fei Huang, Lei Liang, Zhiqiang Zhang, Xiaowei Zhu, Jun Zhou, and Huajun Chen. A comprehensive study of knowledge editing for large language models, 2024. URL https://arxiv.org/abs/2401.01286.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models, 2020. URL https://arxiv.org/abs/2012.00363.

A Details of the AlphaEdit Update Formula

A.1 Closed-Form Solution

The primary objective is to find $\mathbf{\Delta}$ that incorporates new knowledge while preserving both the original model's knowledge and knowledge from previous edits. The optimization problem is formulated as follows:

$$\mathbf{\Delta} = \arg\min_{\tilde{\mathbf{\Delta}}} \left(\|\tilde{\mathbf{\Delta}}\mathbf{P}\mathbf{K} - \mathbf{R}\|^2 + \|\tilde{\mathbf{\Delta}}\mathbf{P}\|^2 + \|\tilde{\mathbf{\Delta}}\mathbf{P}\mathbf{K}_p\|^2 \right)$$

where the three terms represent to the insertion of new information, a regularization term for stable convergence, and the preservation of prior edits, respectively.

This objective has a closed-form solution. The final update matrix $\mathbf{\Delta}' = \mathbf{\Delta}\mathbf{P}$ is given by:

$$\mathbf{\Delta}' = \mathbf{R}\mathbf{K}^\top\mathbf{P}\left(\mathbf{K}_p\mathbf{K}_p^\top\mathbf{P} + \mathbf{K}\mathbf{K}^\top\mathbf{P} + \mathbf{I}\right)^{-1}$$

A.2 Computation of the Projection Matrix $\mathbf{P}$

The matrix $\mathbf{P}$ is a projection matrix designed to project the update $\mathbf{\Delta}$ into the null space of a large key matrix $\mathbf{K}_0$, which represents a vast collection of the model's existing knowledge. This ensures that the update does not interfere with this preserved knowledge, satisfying $\mathbf{\Delta}\mathbf{P}\mathbf{K}_0 = \mathbf{0}$.

Due to the high dimensionality of $\mathbf{K}_0$, the projection is computed using the much smaller covariance matrix $\mathbf{K}_0\mathbf{K}_0^\top$. The procedure is as follows. First, Singular Value Decomposition (SVD) is performed on the covariance matrix:

$$\{\mathbf{U}, \mathbf{\Lambda}, (\mathbf{U})^\top\} = \text{SVD}(\mathbf{K}_0\mathbf{K}_0^\top)$$

Next, the eigenvectors in $\mathbf{U}$ (which are its columns) corresponding to near-zero eigenvalues are identified. A submatrix $\tilde{\mathbf{U}}$ is then constructed using only these selected eigenvectors. Finally, the projection matrix $\mathbf{P}$ is defined as:

$$\mathbf{P} = \tilde{\mathbf{U}}(\tilde{\mathbf{U}})^\top$$

A.3 Computation of the Prior Keys Matrix $\mathbf{K}_p$

The matrix $\mathbf{K}_p$ is used in sequential editing tasks to protect the knowledge updated in previous steps from being disrupted by the current edit. It is constructed by aggregating the key matrices from all prior edits.

Specifically, if there have been $t-1$ previous edits, with corresponding key matrices $\mathbf{K}_1, \mathbf{K}_2, \dots, \mathbf{K}_{t-1}$, then $\mathbf{K}_p$ is the horizontal concatenation of these matrices:

$$\mathbf{K}_p = [\mathbf{K}_1, \mathbf{K}_2, \dots, \mathbf{K}_{t-1}]$$

For the very first edit, $\mathbf{K}_p$ is an empty matrix.

B The Regularization Term $\mathcal{R}$

The optimization objective to find the residual vector δ is given by:

$$\delta = \arg\min_{\tilde{\delta}} \left\{ -\log p\Big(o^* \mid \mathbf{h}^* \leftarrow \mathbf{h} + \tilde{\delta}\Big) + \mathcal{R} \right\}.$$

The regularization term $\mathcal{R}$ is introduced to prevent the model from overfitting to the new fact, which could corrupt existing knowledge. It consists of two components: a KL divergence term and a weight decay term. The full regularization term is formulated as:

$$\mathcal{R} = \lambda_{\text{KL}} D_{\text{KL}}(p(\mathbf{h}) \,||\, p(\mathbf{h} + \tilde{\delta})) + \lambda_{\text{WD}} ||\tilde{\delta}||_2^2.$$

KL Divergence. The first term uses the Kullback-Leibler (KL) divergence to preserve knowledge related to the subject of the edit. This is achieved by computing the divergence on a prompt, such as *"{subject} is a"* (e.g., *"Chrome is a"*). Specifically, it measures the divergence between the output probability distribution from the original hidden state $\mathbf{h}$ and the distribution from the modified hidden state $\mathbf{h} + \tilde{\delta}$. The hyperparameter λ_{KL} controls the strength of this penalty.

Weight Decay. The second term is a weight decay penalty on the L2 norm of the residual vector $\tilde{\delta}$. This term encourages the model to find a smaller solution for $\tilde{\delta}$. The hyperparameter λ_{KL} controls the strength of this penalty.

C The Additive Update

The updated residual stream $\mathbf{h}^*$ is computed via a simple additive update:

$$\mathbf{h}^* = \mathbf{h} + \delta',$$

where our residual vector δ' is defined as:

$$\delta' = (\mathbf{h}^\top \mathbf{w}_2 - \mathbf{h}^\top \mathbf{w}_1)\mathbf{w}_1 + (\mathbf{h}^\top \mathbf{w}_1 - \mathbf{h}^\top \mathbf{w}_2)\mathbf{w}_2.$$

This formulation is simplified by ignoring interactions between the feature directions $\mathbf{w}_1$ and $\mathbf{w}_2$. By assuming interactions between them to be zero (i.e., $\mathbf{w}_1^\top \mathbf{w}_2 = 0$), we can achieve the intended swap of magnitudes with a straightforward additive operation.

To verify that this update swaps the magnitudes of $\mathbf{h}$ along $\mathbf{w}_1$ and $\mathbf{w}_2$, we can compute the new projections $(\mathbf{h}^*)^\top \mathbf{w}_1$ and $(\mathbf{h}^*)^\top \mathbf{w}_2$. Since $\mathbf{w}_1$ and $\mathbf{w}_2$ are unit vectors, $\mathbf{w}_1^\top \mathbf{w}_1 = 1$ and $\mathbf{w}_2^\top \mathbf{w}_2 = 1$.

First, let's compute the projection of $\mathbf{h}^*$ onto $\mathbf{w}_1$:

$$\begin{aligned}
(\mathbf{h}^*)^\top \mathbf{w}_1 &= \big(\mathbf{h} + (\mathbf{h}^\top \mathbf{w}_2 - \mathbf{h}^\top \mathbf{w}_1)\mathbf{w}_1 + (\mathbf{h}^\top \mathbf{w}_1 - \mathbf{h}^\top \mathbf{w}_2)\mathbf{w}_2\big)^\top \mathbf{w}_1 \\
&= \mathbf{h}^\top \mathbf{w}_1 + (\mathbf{h}^\top \mathbf{w}_2 - \mathbf{h}^\top \mathbf{w}_1)(\mathbf{w}_1^\top \mathbf{w}_1) + (\mathbf{h}^\top \mathbf{w}_1 - \mathbf{h}^\top \mathbf{w}_2)(\mathbf{w}_2^\top \mathbf{w}_1) \\
&= \mathbf{h}^\top \mathbf{w}_1 + (\mathbf{h}^\top \mathbf{w}_2 - \mathbf{h}^\top \mathbf{w}_1)(1) + (\mathbf{h}^\top \mathbf{w}_1 - \mathbf{h}^\top \mathbf{w}_2)(0) \\
&= \mathbf{h}^\top \mathbf{w}_1 + \mathbf{h}^\top \mathbf{w}_2 - \mathbf{h}^\top \mathbf{w}_1 \\
&= \mathbf{h}^\top \mathbf{w}_2
\end{aligned}$$

As shown, the new magnitude of the hidden state along $\mathbf{w}_1$ is equal to its original magnitude along $\mathbf{w}_2$.

Next, we compute the projection of $\mathbf{h}^*$ onto $\mathbf{w}_2$:

$$\begin{aligned}(\mathbf{h}^*)^\top \mathbf{w}_2 &= \left(\mathbf{h} + (\mathbf{h}^\top \mathbf{w}_2 - \mathbf{h}^\top \mathbf{w}_1)\mathbf{w}_1 + (\mathbf{h}^\top \mathbf{w}_1 - \mathbf{h}^\top \mathbf{w}_2)\mathbf{w}_2\right)^\top \mathbf{w}_2 \\ &= \mathbf{h}^\top \mathbf{w}_2 + (\mathbf{h}^\top \mathbf{w}_2 - \mathbf{h}^\top \mathbf{w}_1)(\mathbf{w}_1^\top \mathbf{w}_2) + (\mathbf{h}^\top \mathbf{w}_1 - \mathbf{h}^\top \mathbf{w}_2)(\mathbf{w}_2^\top \mathbf{w}_2) \\ &= \mathbf{h}^\top \mathbf{w}_2 + (\mathbf{h}^\top \mathbf{w}_2 - \mathbf{h}^\top \mathbf{w}_1)(0) + (\mathbf{h}^\top \mathbf{w}_1 - \mathbf{h}^\top \mathbf{w}_2)(1) \\ &= \mathbf{h}^\top \mathbf{w}_2 + \mathbf{h}^\top \mathbf{w}_1 - \mathbf{h}^\top \mathbf{w}_2 \\ &= \mathbf{h}^\top \mathbf{w}_1\end{aligned}$$

Similarly, the new magnitude along $\mathbf{w}_2$ becomes the original magnitude along $\mathbf{w}_1$. Thus, this simple additive update, which ignores interactions between the two directions, effectively swaps the magnitudes as intended.

D Experiment Details

D.1 Baseline Methods

Below we provide brief descriptions of the baseline methods used for comparison. Our main set comprises Fine-Tuning (FT; FT-L/FT-W) (Zhu et al., 2020), MEND (Mitchell et al., 2022), PMET (Li et al., 2024), MEMIT (Meng et al., 2023b), and AlphaEdit (Fang et al., 2025). Additional baselines include ROME (Meng et al., 2023a), RECT (Gu et al., 2024), PRUNE (Ma et al., 2024), and NSE (Jiang et al., 2024).

Fine-Tuning (FT-L & FT-W). *FT-L* fine-tunes only the weights of a specific layer (as identified by ROME), rather than all layers. *FT-W* is a variant of FT-L that differs slightly in the loss used for parameter optimization under regularization.

MEND (Model Editor Networks with Gradient Decomposition). Edits large pre-trained models from a single input–output pair by applying a low-rank decomposition to the fine-tuning gradient and using small auxiliary "editor" networks for fast, localized parameter updates that mitigate overfitting.

PMET (Precise Model Editing in Transformers). Observes that hidden states arise from FFN, MHSA, and residual paths. It assumes MHSA encodes general extraction patterns and need not be altered; PMET jointly optimizes hidden states for FFN/MHSA but updates only FFN weights using the optimized FFN state to make more precise edits.

MEMIT (Mass-Editing Memory in a Transformer). Extends ROME to insert many new factual memories efficiently by targeting transformer modules that causally mediate factual recall, enabling simultaneous updates for thousands of associations.

AlphaEdit. Within the locate–then–edit paradigm, projects the parameter perturbation onto the null space of knowledge to be preserved before applying it, so outputs for preserved queries remain unchanged and corruption during sequential edits is reduced.

ROME (Rank-One Model Editing). Identifies key mid-layer feed-forward activations that influence factual predictions and applies a direct rank-one weight update to modify specific factual associations.

RECT (Regularizing Causal Tracing). Regularizes weight updates during editing to prevent excessive changes and overfitting, mitigating side effects (e.g., reasoning degradation) while maintaining general capabilities.

PRUNE (Preserving Representations through Unitary Nullspace Editing). Constrains the edited matrix (e.g., via condition-number control and null-space restrictions) so perturbations remain limited to stored knowledge, preserving overall ability under sequential edits.

D.2 Hyperparameter Settings

We adopt a unified configuration across *locate–then–edit* methods (ROME, MEMIT, PMET, RECT, PRUNE, NSE) as well as AlphaEdit and **SUIT**, and only deviate where method-specific constraints apply. Unless noted otherwise, we set `v_weight_decay`= 0.5, and `kl_factor`= 0.0625. For model-specific layer selections, we use indices {4, 5, 6, 7, 8} for Llama3-8B-Insctruct, {3, 4, 5, 6, 7, 8} for Gpt-J-6b, and {4, 5, 6, 7, 8} for Qwen2.5-7B-Instruct; an exception is ROME, which always edits a single target layer (index 5). **SUIT** does not use `v_weight_decay` or `kl_factor`. In addition, both AlphaEdit and **SUIT** employ `nullspace_threshold`= 2×10^{-2} and `L2`= 10. For **SUIT**, we fix the method-internal hyperparameters $\tau_{\text{energy}} = 0.4$ and $\lambda = 0.3$ for all models. MEND and FT-L/FT-W are tuned following their original papers. Hyperparameter choices follow those specified in the original papers; when not provided, we default to the EasyEdit open-source settings Wang et al. (2024a).

E Detailed Description of Evaluation Metrics

Let o denote the old object and o^* the new object. For each item i, let x_i be the rewrite prompt (i.e., (s_i, r_i)), $N(x_i)$ its paraphrase prompts, and $O(x_i)$ its neighborhood prompts. All probabilities $P_{f_\theta}(\cdot \mid \cdot)$ are computed under the language model f_θ. These evaluation metrics are not new; we follow established practice from prior work (Fang et al., 2025; Meng et al., 2023a;b).

E.1 CounterFact Metrics

Probability-based criterion. The following three metrics use probability comparisons between the edited target o^* and the original o.

Efficacy.

$$\mathbb{E}_i \; \mathbf{1}\Big[P_{f_\theta}\big(o^* \mid x_i\big) \; > \; P_{f_\theta}\big(o \mid x_i\big)\Big].$$

Generalization.

$$\mathbb{E}_i \; \mathbf{1}\Big[P_{f_\theta}\big(o^* \mid N(x_i)\big) \; > \; P_{f_\theta}\big(o \mid N(x_i)\big)\Big].$$

Specificity.

$$\mathbb{E}_i \; \mathbf{1}\Big[P_{f_\theta}\big(o \mid O(x_i)\big) \; > \; P_{f_\theta}\big(o^* \mid O(x_i)\big)\Big].$$

Generation-based criterion (exact match). Let $\tau(o^*) = (o^*_1, \ldots, o^*_{T^*})$. Success if every target token is the greedy choice at its step:

$$\mathbb{E}_i \; \mathbf{1}\left[\forall\, t \in \{1, \ldots, T^*\} : \; o^*_t \; = \; \arg\max_y \; P_{f_\theta}\big(y \,\big|\, o^*_{<t}, \, x_i\big)\right].$$

Fluency (generation entropy) Measure for excessive repetition in model outputs. It uses the entropy of n-gram distributions:

$$-\frac{2}{3} \sum_k g_2(k) \, \log_2 g_2(k) \; + \; \frac{4}{3} \sum_k g_3(k) \, \log_2 g_3(k), \tag{22}$$

where $g_n(\cdot)$ is the n-gram frequency distribution.

Consistency (reference score) The consistency of the model's outputs is evaluated by giving the model f_θ a subject s and computing the cosine similarity between the TF-IDF vectors of the model-generated text and a reference Wikipedia text about o.

E.2 zsRE Metrics

Token-level partial credit. For target string y with tokenization $\tau(y) = (y_1, \ldots, y_{|y|})$ and prompt x, define the token-level accuracy under teacher-forced greedy decoding as

$$\mathrm{TokenAcc}(x, y) \;=\; \frac{1}{|y|} \sum_{t=1}^{|y|} \mathbf{1}\Big[\, y_t \;=\; \arg\max_{v} \; P_{f_\theta}\big(v \,\big|\, y_{<t},\, x\big) \Big] \,.$$

Efficacy. Average token-level accuracy on rewrite prompts:

$$\mathbb{E}_i \big[\, \mathrm{TokenAcc}\big(x_i,\, o^*\big) \,\big] .$$

Generalization. Average token-level accuracy on paraphrase prompts:

$$\mathbb{E}_i \big[\, \mathrm{TokenAcc}\big(N(x_i),\, o^*\big) \,\big] .$$

Specificity. Here, o denotes the ***first token*** generated by the pre-edit model for $O(x_i)$ rather than the dataset's gold answer; we adopt this choice because the language model often does not reproduce the zsRE-provided o. Average token-level accuracy on neighborhood prompts:

$$\mathbb{E}_i \big[\, \mathrm{TokenAcc}\big(O(x_i),\, o\big) \,\big] .$$

F More Experimental Results

F.1 Extended Baseline Comparisons

	Method	Counterfact					
		Eff. ↑	Gen. ↑	Spe. ↑	Flu. ↑	Con. ↑	GC ↑
LLaMA3	Pre-edit	0.0	0.0	100.0	634.9	20.9	62.3
	FT-W	4.0	2.9	43.5	**634.4**	21.4	60.5
	ROME	0.0	0.2	0.0	481.5	4.2	0.0
	RECT	81.6	72.4	36.1	**634.4**	35.3	60.4
	PRUNE	43.3	39.3	17.7	590.4	33.9	45.0
	NSE	1.4	5.8	62.2	609.6	23.1	60.9
	SUIT	**99.7**	**90.3**	**74.2**	631.2	**38.2**	**61.8**
GPT -J	Pre-edit	0.0	0.0	100.0	621.1	23.9	18.6
	FT-W	12.3	2.4	49.0	613.4	25.7	**36.1**
	ROME	0.1	0.2	0.2	407.2	4.2	0.0
	RECT	92.9	85.8	44.4	**625.0**	47.6	14.9
	PRUNE	51.8	56.0	16.9	504.6	29.4	29.4
	NSE	0.8	12.6	54.1	608.0	34.2	27.5
	SUIT	**98.6**	**93.3**	**64.1**	619.4	**49.4**	17.8
Qwen2.5	Pre-edit	0.0	0.0	100.0	625.5	21.9	20.8
	FT-W	47.9	31.7	4.2	476.7	4.5	0.0
	ROME	51.6	33.7	14.2	440.1	15.6	0.6
	RECT	86.3	85.9	42.3	625.8	**37.7**	**59.9**
	PRUNE	28.2	30.7	7.7	588.1	30.4	6.0
	NSE	0.0	0.0	**99.5**	625.6	21.7	39.3
	SUIT	**99.5**	**86.8**	74.4	**626.2**	37.4	23.7

Table 6: Results on Counterfact same setting with Table 1. Abbreviations: Eff. = Efficacy, Gen. = Generalization, Spe. = Specificity, Flu. = Fluency, Con. = Consistency, GC = General Capability.

F.2 Probability-based criterion results

Table 7 presents the results evaluated under the probability-based criterion.

LLama (Prob)			
Method	Eff.↑	Gen.↑	Spe.↑
Pre Edit	0.0	0.0	100.0
FT-W	9.5	6.7	92.4
ROME	67.1	65.5	49.4
RECT	94.0	87.2	73.9
PRUNE	76.5	75.1	59.7
NSE	83.7	53.1	**98.0**
FT-L	93.0	89.1	35.8
MEND	52.7	53.1	48.4
MEMIT	90.8	88.8	71.1
PMET	82.7	81.0	67.0
AlphaEdit	99.7	**94.1**	72.7
SUIT	**100.0**	90.8	84.4

GPT-J (Prob)			
Method	Eff.↑	Gen.↑	Spe.↑
Pre Edit	0.0	0.0	100.0
FT-W	21.8	6.1	88.4
ROME	49.6	48.9	55.6
RECT	98.5	92.8	74.0
PRUNE	87.2	88.3	53.9
NSE	88.5	70.6	**90.9**
FT-L	91.1	78.3	40.3
MEND	46.0	46.1	53.9
MEMIT	97.9	96.8	67.8
PMET	93.7	94.5	68.2
AlphaEdit	**99.6**	**97.9**	78.6
SUIT	99.3	96.2	89.7

Qwen (Prob)			
Method	Eff.↑	Gen.↑	Spe.↑
Pre Edit	0.0	0.0	100.0
FT-W	82.3	71.7	34.8
ROME	76.7	69.2	56.8
RECT	95.9	85.9	42.3
PRUNE	72.2	73.6	56.7
NSE	0.0	0.0	**100.0**
FT-L	82.3	71.7	34.8
MEND	54.0	53.9	46.0
MEMIT	92.2	95.4	75.8
PMET	85.8	86.9	62.1
AlphaEdit	99.2	**98.3**	80.3
SUIT	**100.0**	94.4	95.0

Table 7: Probability-based criterion results (Eff./Gen./Spe.) on three models.

F.3 Detailed F1 Scores on general capability Benchmarks

F.3.1 General Capability Benchmark Datasets

To evaluate the general capabilities of language models, several well-known benchmark datasets are utilized. The **GLUE** (General Language Understanding Evaluation) benchmark is a prominent collection of diverse natural language understanding tasks (Wang et al., 2018). Key datasets included in GLUE are: **SST-2** (The Stanford Sentiment Treebank), a single-sentence classification task for sentiment analysis of movie reviews (Socher et al., 2013); **MRPC** (Microsoft Research Paraphrase Corpus), which involves determining if a pair of sentences are semantically equivalent (Dolan & Brockett, 2005); **RTE** (Recognizing Textual Entailment), a task that assesses whether a premise sentence logically entails a hypothesis (Bentivogli et al., 2009); and **CoLA** (Corpus of Linguistic Acceptability), where the task is to decide if a sentence is grammatically acceptable (Warstadt et al., 2019). Furthermore, **NLI** (Natural Language Inference) tasks, which require inferring the logical relationship (entailment, contradiction, or neutral) between a pair of sentences, are a crucial part of the evaluation (Williams et al., 2018).

Beyond GLUE, more comprehensive benchmarks exist to measure multi-task proficiency. **MMLU** (Massive Multi-task Language Understanding) is a benchmark designed to measure a text model's multi-task accuracy under zero-shot and few-shot settings across a wide range of subjects (Hendrycks et al., 2021a).

F.3.2 General Capability Benchmark Results

Table 8 reports the F1 scores for each benchmark in the General Capability evaluation.

F.4 Evaluation on CHED (Contextual Hop Editing Dataset)

CHED (Park et al., 2025) extends the CounterFact by evaluating whether knowledge edits remain robust under additional prefix contexts. Specifically, each rewrite prompt (s, r) is preceded by sentences derived from either the original subject s, the old object o, the new object o^*, or their one-hop neighbors. The six context types thus test whether the edited model can maintain correctness when auxiliary but semantically related cues are introduced. As shown in Table 9, our method consistently outperforms across all context types, indicating strong resilience to contextual variation.

F.5 Detailed Results for the 5,000-Edit Setting on CounterFact

The table 10 presents the performance metrics on the CounterFact dataset, where 5,000 cases were sequentially edited in batches of 100.

Figure 1 reports the average performance across all benchmarks, while Figure 6 presents the F1 scores for each benchmark individually.

Model	Method	SST	MMLU	MRPC	COLA	RTE	NLI	Avg.
Llama	Pre Edit	81.78	59.93	65.28	76.72	27.65	69.27	63.44
	FT-L	0.00	0.00	37.34	0.00	0.00	0.00	6.22
	MEND	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MEMIT	75.28	55.93	$\underline{64.80}$	69.57	31.46	$\underline{67.90}$	60.82
	PMET	64.25	28.33	60.70	55.52	**32.16**	59.55	50.09
	AlphaEdit	**77.87**	$\underline{57.82}$	61.72	$\underline{76.36}$	$\underline{31.52}$	67.64	$\underline{62.16}$
	SUIT	$\underline{77.18}$	**58.93**	**65.64**	**77.63**	29.53	**69.26**	**63.03**
GPT-J	Pre Edit	0.00	5.78	23.16	21.41	42.67	52.78	24.30
	FT-L	0.00	**17.74**	$\underline{19.67}$	$\underline{15.92}$	**46.13**	45.79	**24.21**
	MEND	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MEMIT	0.00	5.89	6.73	13.29	41.56	**53.33**	20.13
	PMET	0.00	7.81	3.63	12.22	44.69	46.17	19.09
	AlphaEdit	0.00	4.30	5.81	9.26	$\underline{44.91}$	$\underline{52.79}$	19.51
	SUIT	0.00	$\underline{9.44}$	**24.20**	**21.78**	33.66	33.03	$\underline{20.35}$
Qwen	Pre Edit	13.66	2.46	53.32	23.36	11.21	69.23	28.87
	FT-L	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MEND	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	MEMIT	$\underline{54.31}$	0.79	**55.82**	$\underline{24.85}$	11.49	$\underline{61.48}$	**34.79**
	PMET	15.15	**14.78**	2.27	3.07	**27.16**	26.50	14.82
	AlphaEdit	**59.25**	0.39	39.63	6.47	$\underline{11.56}$	51.52	28.14
	SUIT	19.07	$\underline{3.23}$	$\underline{55.03}$	**32.27**	8.80	**66.23**	$\underline{30.77}$

Table 8: F1 scores per benchmark.

Table 9: Performance comparison on CHED. Each column corresponds to a rewrite prompt augmented with one of six prefix–context types: *Subject*, *Obj-Old*, *Obj-New*, and their 1-hop variants.

Context Types	Subject	Obj-Old	Obj-New	Subject Hop	Obj-Old Hop	Obj-New Hop	Avg.
MEMIT	75.4	73.6	77.8	74.1	70.4	75.7	74.2
AlphaEdit	$\underline{92.7}$	$\underline{88.6}$	$\underline{94.2}$	$\underline{90.4}$	$\underline{87.9}$	**92.0**	$\underline{91.0}$
SUIT	**95.7**	**92.0**	**95.6**	**94.3**	**91.2**	$\underline{93.4}$	**93.4**

Table 10: Full performance metrics on the CounterFact dataset. Metrics are grouped by generation-based correctness (Gen) and output probabilities (Prob). The overall score S is the harmonic mean of Efficacy (Eff.), Generalization (Gen.), and Specificity (Spe.). Fluency (N-gram Entropy) and Consistency (Reference Score) values are scaled by 100. Best and second-best results are in **bold** and $\underline{\text{underlined}}$, respectively.

	Method	Generation-based				Probability-based				Fluency ↑	Consistency ↑
		S	Eff. ↑	Gen. ↑	Spe. ↑	S	Eff. ↑	Gen. ↑	Spe. ↑		
Llama	SUIT	**38.0**	**95.8**	58.2	**19.5**	**89.9**	**99.0**	89.0	**83.2**	**624.1**	33.5
	AlphaEdit	29.0	89.7	**69.4**	12.8	80.6	97.6	**92.9**	61.7	613.9	**33.6**

G Detailed Visualization of Component Effects

As stated in § 6.2.2, we analyzed the individual roles of $\mathbf{w}_1$ and $\mathbf{w}_2$ by decomposing our residual vector δ' into its components:

$$\Delta\mathbf{w}_1 = (\mathbf{h}^\top\mathbf{w}_2 - \mathbf{h}^\top\mathbf{w}_1)\mathbf{w}_1$$

$$\Delta\mathbf{w}_2 = (\mathbf{h}^\top\mathbf{w}_1 - \mathbf{h}^\top\mathbf{w}_2)\mathbf{w}_2$$

We then observed the changes in logits for the original object o ("Google") and the new object o^* ("Apple") by incrementally adding each component to the residual stream $\mathbf{h}$, scaled by an interpolation factor $k \in [0, 1]$.

Figure 7 provides a full breakdown of these effects for the edit (*"Chrome"*, *"was developed by"*, *"Apple"*). The results confirm our finding that the $\Delta\mathbf{w}_1$ component is effective at increasing the

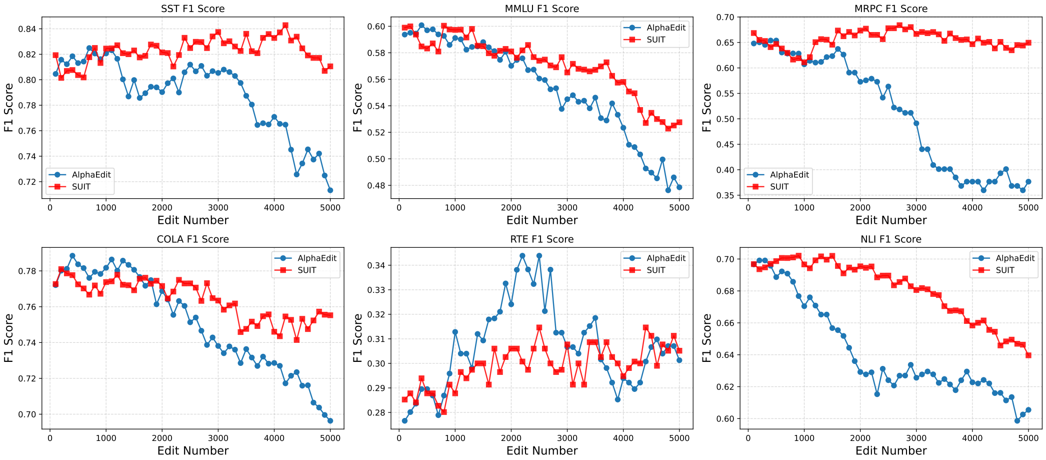


Figure 6: F1 scores for each benchmark.

logit of the new object o^*, while the $\Delta \mathbf{w}_2$ component is effective at decreasing the logit of the old object o.

However, the plots also illustrate that $\mathbf{w}_1$ also suppresses the old object o, and $\mathbf{w}_2$ promotes the new object o^*, rather than each playing only a single role. This visual evidence reinforces the point that the components are not fully disentangled.

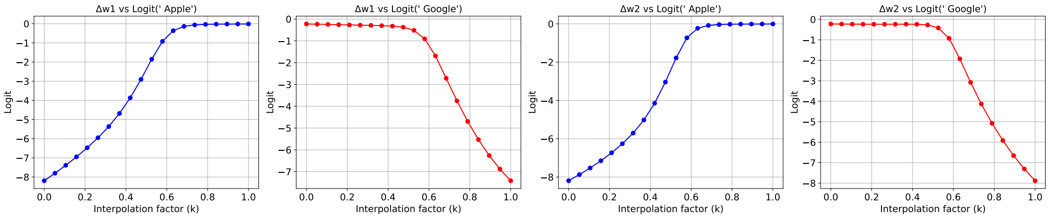
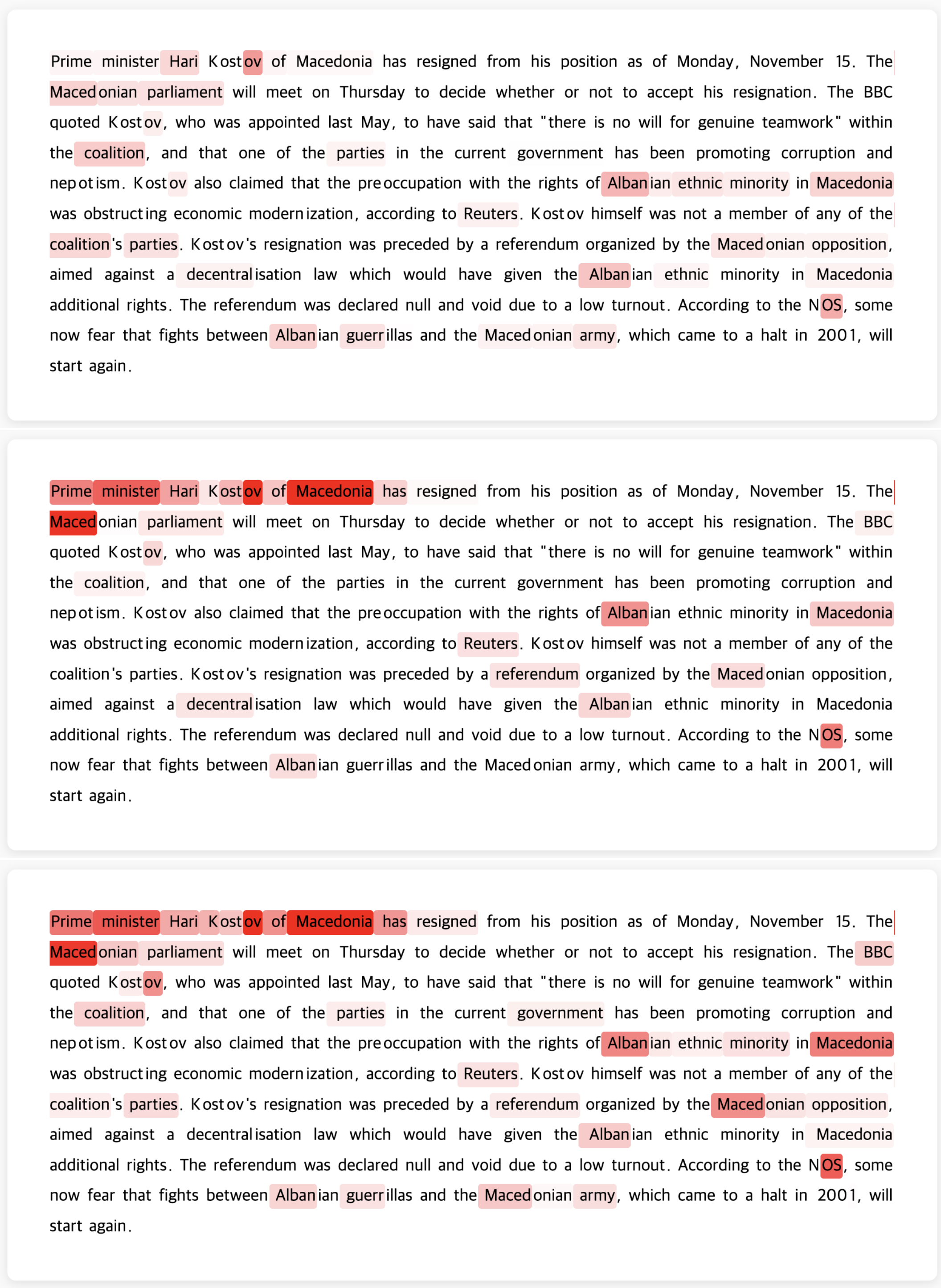


Figure 7: A full breakdown of the effects of applying scaled components $\Delta \mathbf{w}_1$ and $\Delta \mathbf{w}_2$ to the residual stream.

H Additional Heatmap Visualizations

This appendix provides further examples from the Wikinews Article Dataset, illustrating the perturbation at each entity's last-token position. For each of the five articles presented, we show the L_2 norm of the difference in the residual streams of the final edited layer between the original model and models edited using SUIT, MEMIT, and AlphaEdit.

The figures are presented in the order: SUIT, MEMIT, AlphaEdit. As visually represented by the color intensity, SUIT consistently reduces the perturbation on the last token of the subject entity compared to both MEMIT and AlphaEdit. The same color indicates the same amount of perturbation across all figures.

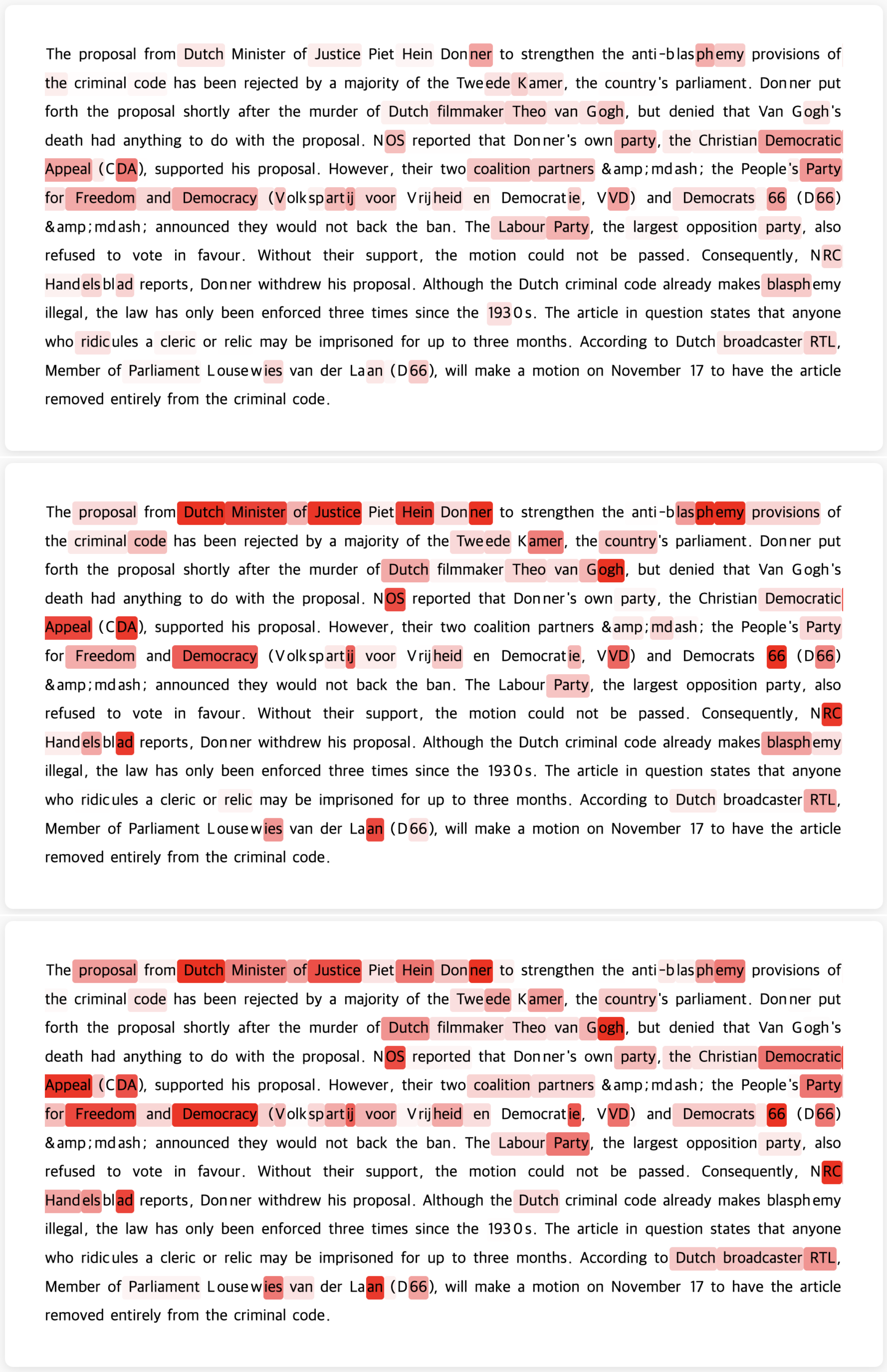


Prime minister Hari Kostov of Macedonia has resigned from his position as of Monday, November 15. The Macedonian parliament will meet on Thursday to decide whether or not to accept his resignation. The BBC quoted Kostov, who was appointed last May, to have said that "there is no will for genuine teamwork" within the coalition, and that one of the parties in the current government has been promoting corruption and nepotism. Kostov also claimed that the preoccupation with the rights of Albanian ethnic minority in Macedonia was obstructing economic modernization, according to Reuters. Kostov himself was not a member of any of the coalition's parties. Kostov's resignation was preceded by a referendum organized by the Macedonian opposition, aimed against a decentralisation law which would have given the Albanian ethnic minority in Macedonia additional rights. The referendum was declared null and void due to a low turnout. According to the NOS, some now fear that fights between Albanian guerrillas and the Macedonian army, which came to a halt in 2001, will start again.

Prime minister Hari Kostov of Macedonia has resigned from his position as of Monday, November 15. The Macedonian parliament will meet on Thursday to decide whether or not to accept his resignation. The BBC quoted Kostov, who was appointed last May, to have said that "there is no will for genuine teamwork" within the coalition, and that one of the parties in the current government has been promoting corruption and nepotism. Kostov also claimed that the preoccupation with the rights of Albanian ethnic minority in Macedonia was obstructing economic modernization, according to Reuters. Kostov himself was not a member of any of the coalition's parties. Kostov's resignation was preceded by a referendum organized by the Macedonian opposition, aimed against a decentralisation law which would have given the Albanian ethnic minority in Macedonia additional rights. The referendum was declared null and void due to a low turnout. According to the NOS, some now fear that fights between Albanian guerrillas and the Macedonian army, which came to a halt in 2001, will start again.

Prime minister Hari Kostov of Macedonia has resigned from his position as of Monday, November 15. The Macedonian parliament will meet on Thursday to decide whether or not to accept his resignation. The BBC quoted Kostov, who was appointed last May, to have said that "there is no will for genuine teamwork" within the coalition, and that one of the parties in the current government has been promoting corruption and nepotism. Kostov also claimed that the preoccupation with the rights of Albanian ethnic minority in Macedonia was obstructing economic modernization, according to Reuters. Kostov himself was not a member of any of the coalition's parties. Kostov's resignation was preceded by a referendum organized by the Macedonian opposition, aimed against a decentralisation law which would have given the Albanian ethnic minority in Macedonia additional rights. The referendum was declared null and void due to a low turnout. According to the NOS, some now fear that fights between Albanian guerrillas and the Macedonian army, which came to a halt in 2001, will start again.

Figure 8: Perturbation heatmaps for sample article 1.

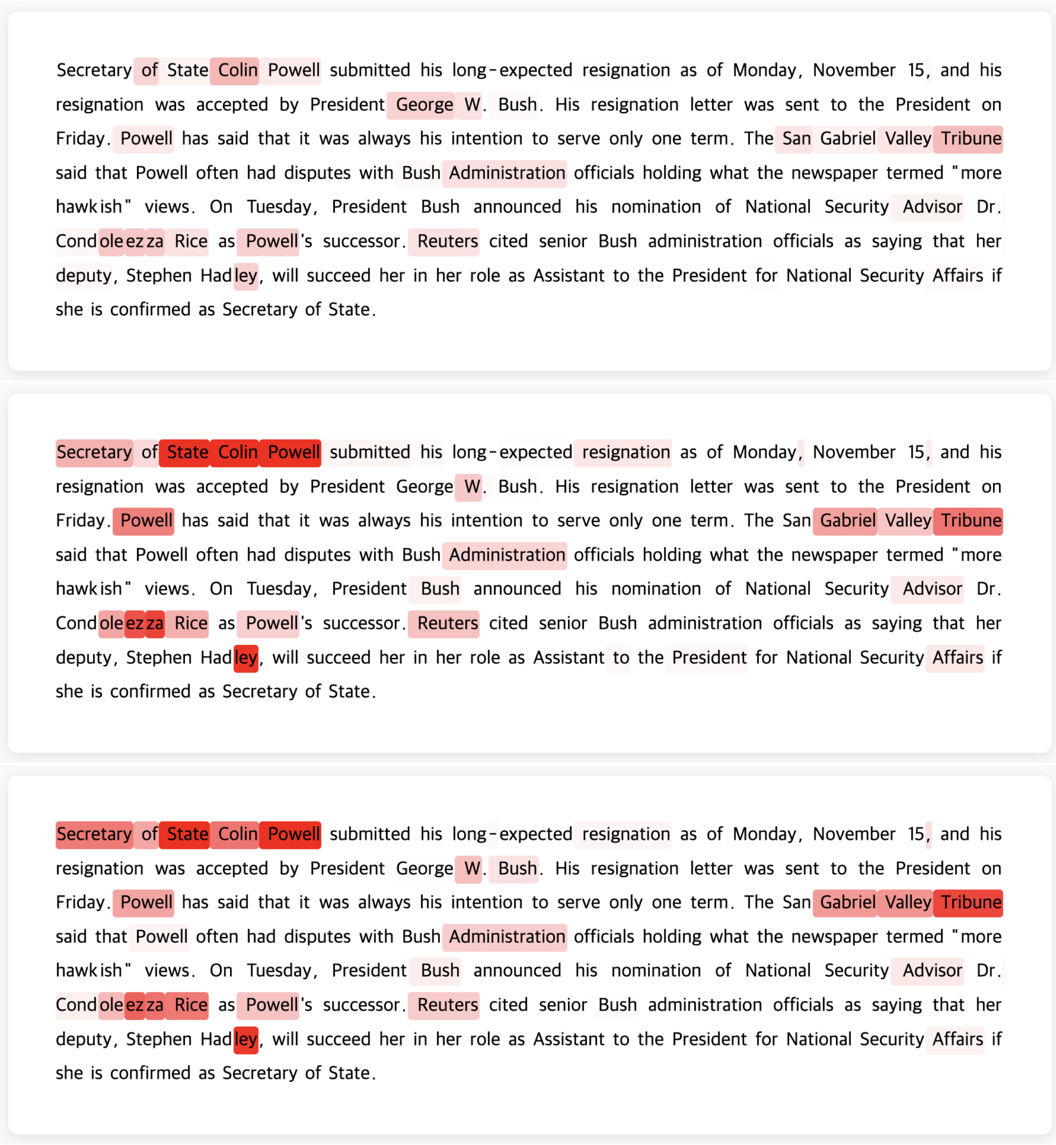


The proposal from Dutch Minister of Justice Piet Hein Donner to strengthen the anti-blasphemy provisions of the criminal code has been rejected by a majority of the Tweede Kamer, the country's parliament. Donner put forth the proposal shortly after the murder of Dutch filmmaker Theo van Gogh, but denied that Van Gogh's death had anything to do with the proposal. NOS reported that Donner's own party, the Christian Democratic Appeal (CDA), supported his proposal. However, their two coalition partners &mdash; the People's Party for Freedom and Democracy (Volkspartij voor Vrijheid en Democratie, VVD) and Democrats 66 (D66) &mdash; announced they would not back the ban. The Labour Party, the largest opposition party, also refused to vote in favour. Without their support, the motion could not be passed. Consequently, NRC Handelsblad reports, Donner withdrew his proposal. Although the Dutch criminal code already makes blasphemy illegal, the law has only been enforced three times since the 1930s. The article in question states that anyone who ridicules a cleric or relic may be imprisoned for up to three months. According to Dutch broadcaster RTL, Member of Parliament Lousewies van der Laan (D66), will make a motion on November 17 to have the article removed entirely from the criminal code.

The proposal from Dutch Minister of Justice Piet Hein Donner to strengthen the anti-blasphemy provisions of the criminal code has been rejected by a majority of the Tweede Kamer, the country's parliament. Donner put forth the proposal shortly after the murder of Dutch filmmaker Theo van Gogh, but denied that Van Gogh's death had anything to do with the proposal. NOS reported that Donner's own party, the Christian Democratic Appeal (CDA), supported his proposal. However, their two coalition partners &mdash; the People's Party for Freedom and Democracy (Volkspartij voor Vrijheid en Democratie, VVD) and Democrats 66 (D66) &mdash; announced they would not back the ban. The Labour Party, the largest opposition party, also refused to vote in favour. Without their support, the motion could not be passed. Consequently, NRC Handelsblad reports, Donner withdrew his proposal. Although the Dutch criminal code already makes blasphemy illegal, the law has only been enforced three times since the 1930s. The article in question states that anyone who ridicules a cleric or relic may be imprisoned for up to three months. According to Dutch broadcaster RTL, Member of Parliament Lousewies van der Laan (D66), will make a motion on November 17 to have the article removed entirely from the criminal code.

The proposal from Dutch Minister of Justice Piet Hein Donner to strengthen the anti-blasphemy provisions of the criminal code has been rejected by a majority of the Tweede Kamer, the country's parliament. Donner put forth the proposal shortly after the murder of Dutch filmmaker Theo van Gogh, but denied that Van Gogh's death had anything to do with the proposal. NOS reported that Donner's own party, the Christian Democratic Appeal (CDA), supported his proposal. However, their two coalition partners &mdash; the People's Party for Freedom and Democracy (Volkspartij voor Vrijheid en Democratie, VVD) and Democrats 66 (D66) &mdash; announced they would not back the ban. The Labour Party, the largest opposition party, also refused to vote in favour. Without their support, the motion could not be passed. Consequently, NRC Handelsblad reports, Donner withdrew his proposal. Although the Dutch criminal code already makes blasphemy illegal, the law has only been enforced three times since the 1930s. The article in question states that anyone who ridicules a cleric or relic may be imprisoned for up to three months. According to Dutch broadcaster RTL, Member of Parliament Lousewies van der Laan (D66), will make a motion on November 17 to have the article removed entirely from the criminal code.

Figure 9: Perturbation heatmaps for sample article 2.

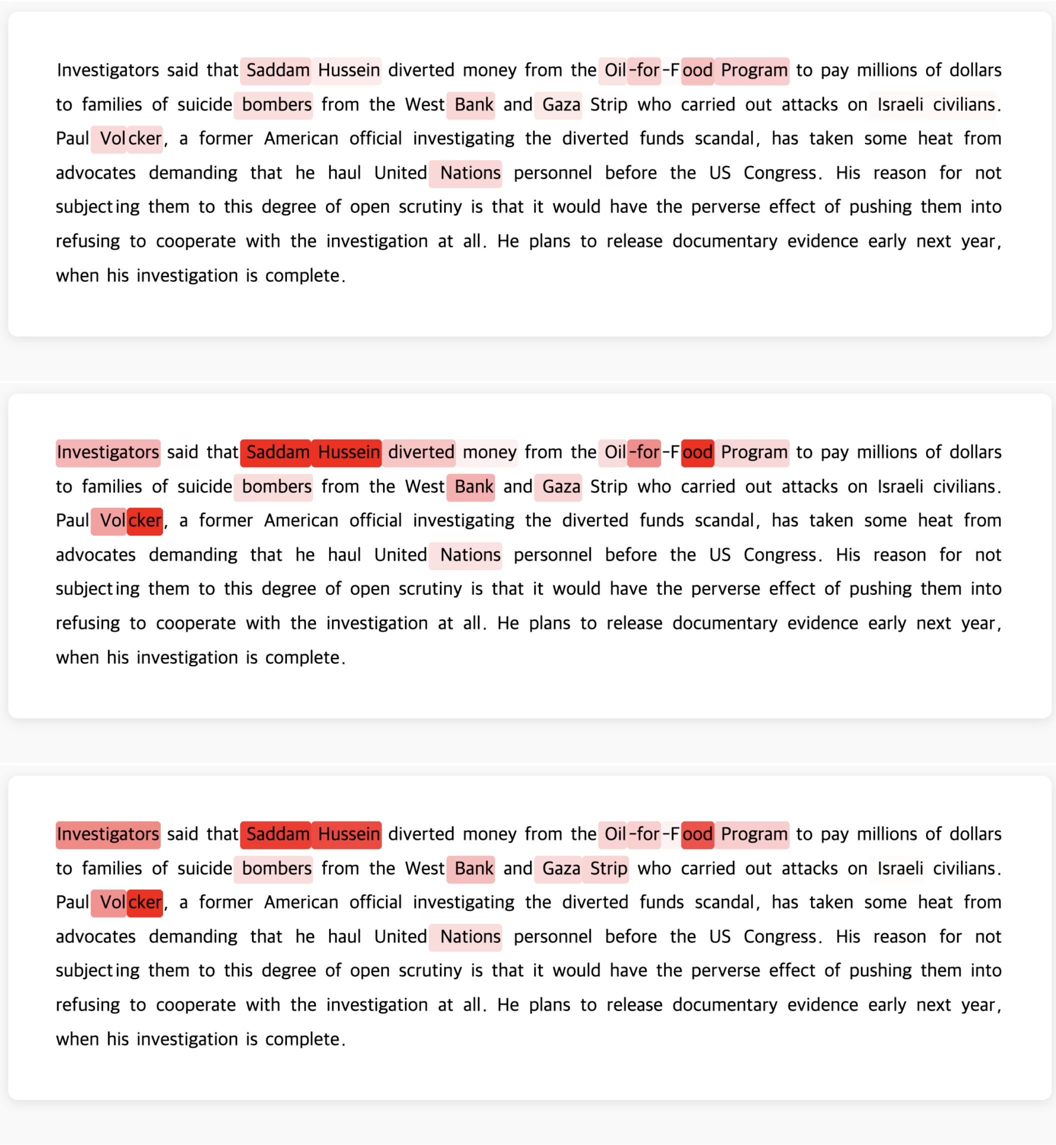


Secretary of State Colin Powell submitted his long-expected resignation as of Monday, November 15, and his resignation was accepted by President George W. Bush. His resignation letter was sent to the President on Friday. Powell has said that it was always his intention to serve only one term. The San Gabriel Valley Tribune said that Powell often had disputes with Bush Administration officials holding what the newspaper termed "more hawkish" views. On Tuesday, President Bush announced his nomination of National Security Advisor Dr. Condoleezza Rice as Powell's successor. Reuters cited senior Bush administration officials as saying that her deputy, Stephen Hadley, will succeed her in her role as Assistant to the President for National Security Affairs if she is confirmed as Secretary of State.

Secretary of State Colin Powell submitted his long-expected resignation as of Monday, November 15, and his resignation was accepted by President George W. Bush. His resignation letter was sent to the President on Friday. Powell has said that it was always his intention to serve only one term. The San Gabriel Valley Tribune said that Powell often had disputes with Bush Administration officials holding what the newspaper termed "more hawkish" views. On Tuesday, President Bush announced his nomination of National Security Advisor Dr. Condoleezza Rice as Powell's successor. Reuters cited senior Bush administration officials as saying that her deputy, Stephen Hadley, will succeed her in her role as Assistant to the President for National Security Affairs if she is confirmed as Secretary of State.

Secretary of State Colin Powell submitted his long-expected resignation as of Monday, November 15, and his resignation was accepted by President George W. Bush. His resignation letter was sent to the President on Friday. Powell has said that it was always his intention to serve only one term. The San Gabriel Valley Tribune said that Powell often had disputes with Bush Administration officials holding what the newspaper termed "more hawkish" views. On Tuesday, President Bush announced his nomination of National Security Advisor Dr. Condoleezza Rice as Powell's successor. Reuters cited senior Bush administration officials as saying that her deputy, Stephen Hadley, will succeed her in her role as Assistant to the President for National Security Affairs if she is confirmed as Secretary of State.

Figure 10: Perturbation heatmaps for sample article 3.

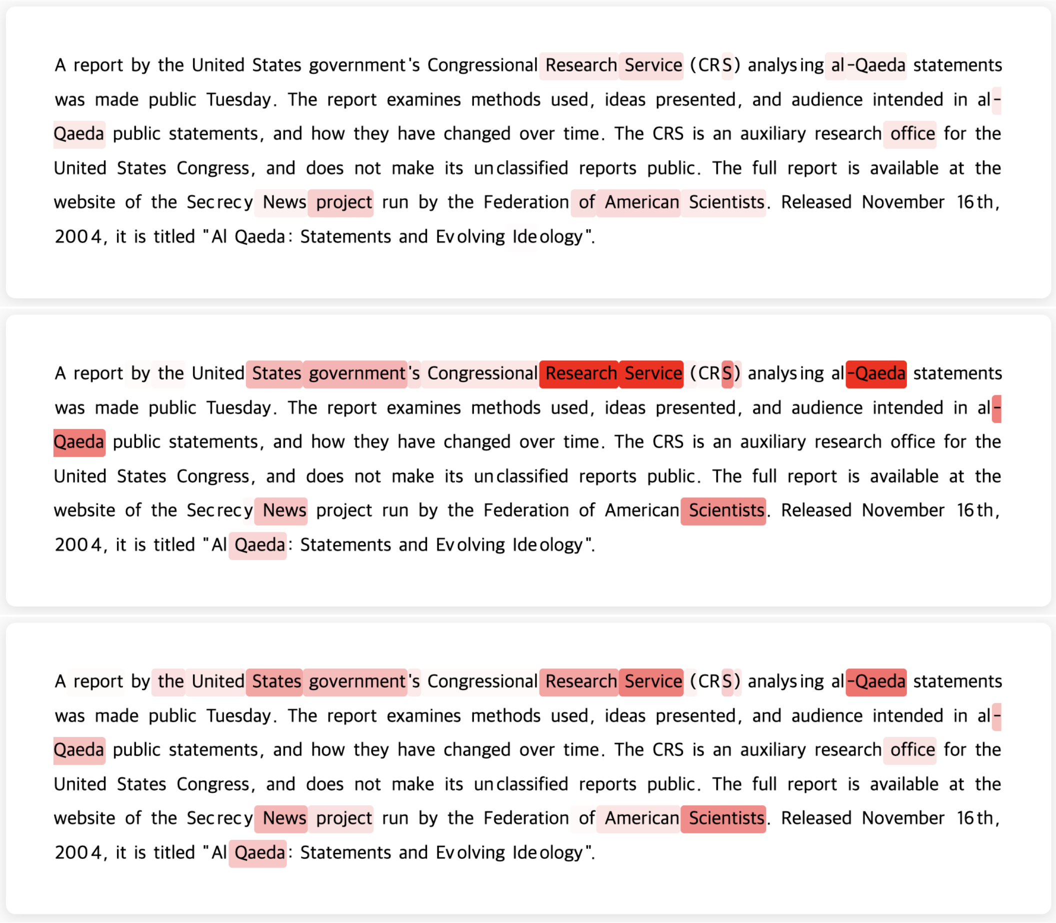


Investigators said that Saddam Hussein diverted money from the Oil-for-Food Program to pay millions of dollars to families of suicide bombers from the West Bank and Gaza Strip who carried out attacks on Israeli civilians. Paul Volcker, a former American official investigating the diverted funds scandal, has taken some heat from advocates demanding that he haul United Nations personnel before the US Congress. His reason for not subjecting them to this degree of open scrutiny is that it would have the perverse effect of pushing them into refusing to cooperate with the investigation at all. He plans to release documentary evidence early next year, when his investigation is complete.

Investigators said that Saddam Hussein diverted money from the Oil-for-Food Program to pay millions of dollars to families of suicide bombers from the West Bank and Gaza Strip who carried out attacks on Israeli civilians. Paul Volcker, a former American official investigating the diverted funds scandal, has taken some heat from advocates demanding that he haul United Nations personnel before the US Congress. His reason for not subjecting them to this degree of open scrutiny is that it would have the perverse effect of pushing them into refusing to cooperate with the investigation at all. He plans to release documentary evidence early next year, when his investigation is complete.

Investigators said that Saddam Hussein diverted money from the Oil-for-Food Program to pay millions of dollars to families of suicide bombers from the West Bank and Gaza Strip who carried out attacks on Israeli civilians. Paul Volcker, a former American official investigating the diverted funds scandal, has taken some heat from advocates demanding that he haul United Nations personnel before the US Congress. His reason for not subjecting them to this degree of open scrutiny is that it would have the perverse effect of pushing them into refusing to cooperate with the investigation at all. He plans to release documentary evidence early next year, when his investigation is complete.

Figure 11: Perturbation heatmaps for sample article 4.



A report by the United States government's Congressional Research Service (CRS) analysing al-Qaeda statements was made public Tuesday. The report examines methods used, ideas presented, and audience intended in al-Qaeda public statements, and how they have changed over time. The CRS is an auxiliary research office for the United States Congress, and does not make its unclassified reports public. The full report is available at the website of the Secrecy News project run by the Federation of American Scientists. Released November 16th, 2004, it is titled "Al Qaeda: Statements and Evolving Ideology".

A report by the United States government's Congressional Research Service (CRS) analysing al-Qaeda statements was made public Tuesday. The report examines methods used, ideas presented, and audience intended in al-Qaeda public statements, and how they have changed over time. The CRS is an auxiliary research office for the United States Congress, and does not make its unclassified reports public. The full report is available at the website of the Secrecy News project run by the Federation of American Scientists. Released November 16th, 2004, it is titled "Al Qaeda: Statements and Evolving Ideology".

A report by the United States government's Congressional Research Service (CRS) analysing al-Qaeda statements was made public Tuesday. The report examines methods used, ideas presented, and audience intended in al-Qaeda public statements, and how they have changed over time. The CRS is an auxiliary research office for the United States Congress, and does not make its unclassified reports public. The full report is available at the website of the Secrecy News project run by the Federation of American Scientists. Released November 16th, 2004, it is titled "Al Qaeda: Statements and Evolving Ideology".

Figure 12: Perturbation heatmaps for sample article 5.