Interpretable Catastrophic Forgetting of Large Language Model Fine-tuning via Instruction Vector

Anonymous ACL submission

Abstract

Fine-tuning large language models (LLMs) can cause them to lose their general capabilities. However, the intrinsic mechanisms behind such forgetting remain unexplored. In this paper, we begin by examining this phenomenon by focusing on knowledge understanding and instruction following, with the latter identified as the main contributor to forgetting during fine-tuning. Consequently, we propose the Instruction Vector (IV) framework to capture model representations highly related to specific instruction-following capabilities, thereby making it possible to understand model-intrinsic forgetting. Through the analysis of IV dynamics pre and post-training, we suggest that fine-tuning mostly adds specialized reasoning patterns instead of erasing previous skills, which may appear as forgetting. Building on this insight, we develop IV-guided training, which aims to preserve original computation graph, thereby mitigating catastrophic forgetting. Empirical tests on three benchmarks confirm the efficacy of this new approach, supporting the relationship between IVs and forgetting. Our code will be made available soon.

1 Introduction

Instruction fine-tuning (Peng et al., 2023; Chung et al., 2024) has emerged as an indispensable ingredient in the development of Large Language Models (LLMs) (Brown et al., 2020; Radford et al., 2019; Touvron et al., 2023b),enabling them to meet the demands of specific domains (Roziere et al., 2023; Thirunavukarasu et al., 2023) and human preferences (Ouyang et al., 2022). However, a notable concern with this fine-tuning is "catastrophic forgetting" (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017), where models may lose essential skills (Dou et al., 2023; Chen et al., 2023) such as mathematical reasoning while adjusting to user instructions. This raises questions about which

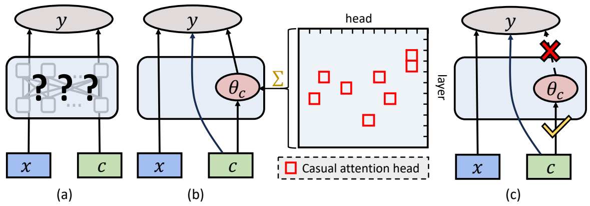


Figure 1: Instruction vector hypothesis for LLM understanding. θ_c is extracted by aggregating representations of attention heads identified to have causal influence to the output. Forgetting is resulted from the suppression of instruction vector associated computation graph.

abilities are most susceptible to forgetting and the underlying causes of these losses in LLMs.

Research on LLM forgetting (Luo et al., 2024; Wang et al., 2023b; Wu et al., 2024a) generally examines changes in abilities like reading comprehension, factual retention, mathematical skills, and code generation, underscoring the existence of catastrophic forgetting. Despite these findings, there is a notable gap in understanding the internal mechanisms responsible for these losses. To date, only a few studies, such as Kotha et al. (2024) proposing the task inference hypothesis, have begun to explore how conflicts between task processors might lead to forgetting. Nevertheless, the literature still lacks comprehensive insights into the exact changes that result in forgetting, leaving open questions about whether these changes involve overwriting of old modules or if they are simply overshadowed by new, specialized patterns.

In this paper, we first present a novel perspective to investigate catastrophic forgetting in LLMs, focusing on the capabilities developed during pretraining and alignment phases. We suggest that the task proficiency in LLMs involves understanding task-specific knowledge and following instructions, assessed through *Knowledge Probability* $P(y|x)$ and *Instruction Probability* $P(y^c|c, x)$, respectively (as depicted in Fig. 2). Our empirical analysis within a continual instruction tuning framework reveals distinct forgetting patterns between these two aspects, with shifts in instruction following primarily driving performance declines.

To investigate the internal changes of the model during forgetting, we introduce the Instruction Vector (IV) framework to extract representations closely associated with the task processing. We hypothesize a straightforward yet robust computational graph for LLMs (see Fig. 1 b), featuring an intermediate variable θ_c crucial for task performance. The presence or absence of θ_c directly impacts the model's capability to handle instruction c. This hypothesis is supported by causal intervention experiments in Sec. 3.2. By analyzing IV dynamics pre and post-training, we find minor changes in IV expression with forgetting happens. Furthermore, explicitly incorporating IV into the model's computational graph can recover the mastery of the corresponding instruction. This results indicate that fine-tuning mostly adds specialized reasoning patterns instead of erasing previous skills, which may appear as forgetting.

Building on these insights, we develop an IV-guided training methodology to mitigate catastrophic forgetting. This method incorporates a progressive IV-intervention training mechanism, in which the IV is initially introduced through intervention and is then gradually phased out during the training process. The deliberate inclusion of IV aids in optimizing the model by ensuring adherence to the IV-related computational graph, thereby minimizing the overshadowing effect of new reasoning pathways. Additionally, we have introduced an IV-based KL-Divergence loss function to reduce the discrepancies between zero-shot and IV-intervened logits, ensuring that the model's behavior remains aligned with the original computational structure. Validated across multiple datasets, this method significantly alleviate forgetting in both general and in-context learning abilities, confirming the link between IV and forgetting.

Main Findings and Contributions. **(1)** We introduce a new perspective on catastrophic forgetting by using Knowledge and Instruction Probability to evaluate how well LLMs retain task-specific knowledge and follow instructions after tuning, showing that changes in instruction adherence mainly drive performance declines. **(2)** We are the first to interpret forgetting with the Instruction Vector framework, identifying inherent changes during fine-tuning. The findings indicate that fine-tuning generally introduces specialized reasoning patterns rather than removing existing skills. **(3)** We develop an IV-guided training approach that focuses on preserving and realigning the model's computational graph during fine-tuning. This significantly enhances the general and in-context learning capabilities across various datasets in continual learning.

2 Catastrophic Forgetting in LLMs

In this section, we present a new perspective to investigate catastrophic forgetting in LLMs, concentrating on the capabilities embedded within pre-training and instruction tuning stages, as opposed to focusing on pure performance shifts as noted in earlier studies (Wang et al., 2023b; Zhai et al., 2023). We start with a discussion on the capabilities encoded in LLMs, proceed to develop continual instruction tuning setup to investigate forgetting, and conclude with the empirical observations.

Let M denote the model pre-trained on large scale data corpus $\mathcal{D}_{PT} = \{\boldsymbol{X}_i\}$ with the language modeling task (Brown et al., 2020; Radford et al., 2019). We assume that M has built an impressive ability to capture world knowledge across various domains, i.e., M assigns the maximum likelihood to $P(y|x, M)$ for certain datasets denoted by $D^K = \{(x_i, y_i)\} \in \mathcal{D}_{PT}$. Here, the pair $[x_i, y_i]$ may represent a segment extracted from raw text $\boldsymbol{X}_j$. For example, consider x being "The capital city of Japan is" and y being "Tokyo"; such a pairing frequently appears in blogs. In this paper, we refer to $P(y|x, M)$ as the ***Knowledge Probability***, which serves as a metric for evaluating the model's proficiency in comprehending world knowledge.

While processing instructional data, the model M is presented with the dataset $D^c = \{(c, x_i, y_i^c)\}$, where each tuple consists of an instruction c, an input prompt x_i, and an expected output y_i^c. For

x	y	c	y^c
The capital city of Japan	Tokyo	Choose the answer from 1. kyooto, 3. okinawa, 2. nara, and 4. tokyo	4
A little girl in a room standing in front of some chairs is hitting a dora pinata. she	hits it a few times and then its someone else's turn	What is the best end in the following. A: "makes an orange drink from a bucket.", B: "hits it a few times and then its someone else's turn."	B
Last item in the list [mint, grateful, vulture, resilient, build] is	build	Translate to spanish	construir

Figure 2: Task in world knowledge form (x, y) and instruction form (x, c, y^c).

instance, c might be "Choose the best answer from A, B, C, and D (with options given).", x could be "The capital city of Japan is", and y^c would be "D", which aligns with the answer "Tokyo". The model is supposed to generate y^c that accurately responds to the instruction c with the context of x, i.e., maximize $P(y^c|c, x, M)$, which is termed as the ***Instruction Probability***.

In this paper, when discussing catastrophic forgetting of a task, we consider alterations in both *Knowledge* and *Instruction Probabilities*. Typically, a test instance x_i is typically presented as a tuple (x_i, y_i, c, y_i^c) (examples are listed in Fig. 2), with shifts in $P(y_i^c|c, x_i, M)$ signaling variations in the model's proficiency in instruction processing and knowledge understanding and shifts in $P(y_i|x_i, M)$ solely reflect changes in the world knowledge comprehension. Our work go beyond simple performance metrics evaluation, offering a detailed examination of distinct capabilities amidst CF. This method reveals if performance degradation stems from an actual loss of world knowledge or a reduction in the ability to follow instructions.

Continual instruction tuning setup. To explore CF in LLMs, we conduct an empirical study within the continual instruction tuning framework. In this setup, a model is sequentially trained on a series of streaming tasks, denoted as $\{D^{c_1}, D^{c_2}, ..., D^{c_T}\}$. Here, $D^{c_t} = \{(c, x_i, y_i^c)\}$ symbolizes the t-th task associated with a specific instruction c_t. While learning each task D^{c_t}, the model can only access to the corresponding data, with the goal of minimizing loss on all learned tasks. Specifically, the model is optimized with $\min_M \frac{1}{N}\sum_{i=1}^{N} \ell(y_i, M(c, x_i))$, where N is the size of training set and ℓ is usually the cross-entropy loss on the entire vocabulary. In addition to avoiding forgetting on previous learned tasks $\{D^{c_1}, ..., D^{c_{t-1}}\}$, the model is also evaluated on held-out evaluation sets (e.g., CommonsenseQA (Talmor et al., 2018), MMLU (Hendrycks et al., 2020)) to measure its general ability.

We select two different continual instruction tuning benchmarks. The first is from TRACE (Wang et al., 2023b) benchmark, which consists of 6 different complex generation tasks including multi-choice QA, code generation, mathematical reasoning and summary. The second is called FUNC, adapted from the datasets in Todd et al. (2023), in which tasks have clear and simple instructions. For example, task Verb-Spanish and Last-Spanish are both translation task but differ in the selection from list. For the general evaluation datasets, we utilize Hellaswag (Zellers et al., 2019), ARC-challenge (Clark et al., 2018), CommonsenseQA (Talmor et al., 2018), and MMLU-social (Hendrycks et al., 2020). The detailed dataset information and evaluation metrics are present in Appendix A.

We adopt LLAMA2-7B-Chat (Touvron et al., 2023b) as the base model, with its effectiveness in both understanding world knowledge and following instructions. Without specific notification, the model is fine-tuned with LORA approach (Hu et al., 2021), using the Adam optimizer with a learning rate set to 1e-4. Additional details regarding the implementation are provided in the Appendix C.

Forgetting properties in knowledge and instruction probabilities. In our empirical study, we aim to investigate the factors responsible for the model performance drop. To show this, we present the accuracy curve for task in knowledge and instruction forms (cases in Fig. 2) during continual tuning in Fig. 3. Knowledge accuracy is determined by evaluating $P(y|x)$, whereas instruction accuracy is derived from $P(y^c|c, x)$. The reported accuracy follows the evaluation method in Brown et al. (2020); Bordes et al. (2016) which involves choosing the label with the highest log-likelihood. The results reveal a consistent presence of the forgetting effect in LLMs across both general and newly acquired tasks throughout continual instruction tuning. More observations are as follow:

1) *Instruction Following Accuracy Decline*. At the end of training sequence, the average instruction accuracy for the general evaluation set decreases by 10.24 as compared to the pre-trained model. On the other hand, knowledge accuracy sees an average increase of 1.93. This suggests loss in instruction following ability is the reason for task performance drop. 2) *In-Context Learning (ICL) Ineffectiveness*: When attempting to recover performance with ICL (see the red line in Fig. 3), we observe a average decrease of 14.67 in performance compared to zero-shot results. The significant decline indicates that the bias in instruction-following ability is further magnified by ICL. 3) *Severe Forgetting of Newly Learned Concepts*: Forgetting of newly acquired skills is particularly significant. The drop in results for Cstance reaches as much as 3.0 points at each stage of training, while in tasks like ARC the number is just 0.63.

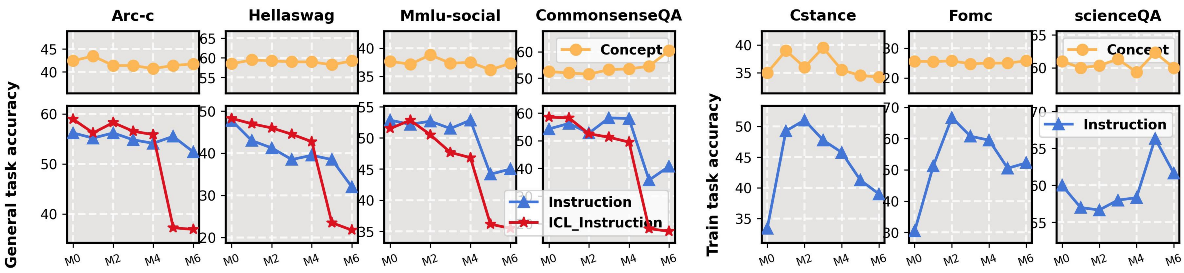


Figure 3: Accuracy curve across naive sequential instruction fine-tuning on the TRACE benchmark. X-axis delineates the stages through training, with "M0" indicating the original pre-trained model, and "Mi" signifying the model post-instruction fine-tuning for the i-th task in sequence. The tasks follow the sequence of Cstance, Fomc, Meetingbank, Py150, ScienceQA, and Numgluecm. Y-axis indicates the rank classification accuracy. Notably, the first four datasets are absent from the training set, whereas the final three datasets are part of the training distribution.

3 Interpret Catastrophic Forgetting via Instruction Vector

Our empirical research indicates that, during the tuning process, models tend to forget instruction-following capabilities as opposed to world knowledge understanding aptitudes. To further investigate the inherent mechanisms of such forgetting, we introduces a framework for interpretability, utilizing Instruction Vectors (IV) to decouple the distinct functionalities of the model. This approach is inspired by the ideas presented by Todd et al. (2023) and Hendel et al. (2023), which suggest that an input-output function can be represented as a vector within LLMs. We reveal that the activation level of IV is positively correlated with the LLMs' proficiency in relevant instruction-following skills during training. Through the analysis of IV's consistency before and after instruction tuning, this paper elucidates the fundamental mechanisms of forgetting within LLMs.

Subsequently, we will first put forth our hypothesis and then introduce the Instruction Vectors framework. Finally, displaying the experimental results on IV, unveiling the dynamic process of forgetting.

3.1 Instruction Vector Hypothesis

Task in instruction dataset D^c is to predict a target variable y_c, given a token sequence x conditioned on instruction c. We assume a potentially high-dimensional latent variable θ_c exists, which governs the model's capability in following instruction c. This suggests a direct computational graph relationship among x, c, θ_c, and y_c, mathematically depicted as $f_M(x, c, \theta_c) \rightarrow y_c$, as illustrated in Fig. 4. Here, f_M denotes the mapping function with model M and we call $f_M(x, c, \theta_c) \rightarrow y_c$ the IV-associate computation graph.

Our hypothesis about the computational graph is supported by key observations illustrated in Fig. 4: i) In (a-c), by intervening zero-shot input inference with representations drawn from in-context learning (ICL) samples (see Sec. 3.2), accuracy improve from 24% to 68%. The effectiveness of this representation aligns with our definition of θ_c, which may be activated by introducing a prompt before input or directly adding to the hidden states during the inference. ii) In (d,e), removing certain representations from well-behaved model results in a dramatic decline in performance from 52% to 0%, indicating a reliance on θ_c for producing y_c, beyond just the inputs x and c. iii) Moreover, the differential impact on task performance in knowledge and instruction form point to a separation in the model's ability to handle x and c. Hence, it's reasonable to conjecture that output relies on θ_c as opposed to $\theta_{x,c}$. Given the focus of this paper on instruction forgetting, the potential influence of θ_x is omitted in the following analysis.

3.2 Instruction Vector

We next consider how to extract θ_c for a given dataset D^c, drawing on the concept of function vectors proposed by Todd et al. (2023). This extraction is carried out using in-context learning (ICL) samples, where the model incorporates task-relevant information into its hidden states as it engages with examples with the ICL prompt. This process is associated with the emergence of θ_c (Todd et al., 2023; Hendel et al., 2023). Subsequently, a causal mediation analysis (Pearl, 2013; Vig et al., 2020; Li et al., 2024) is conducted on the ICL inputs to identify attention heads with significant causal impacts on the output, and aggregating their representations results in θ_c. Interestingly, this vector remains effective even under zero-shot input scenarios, as

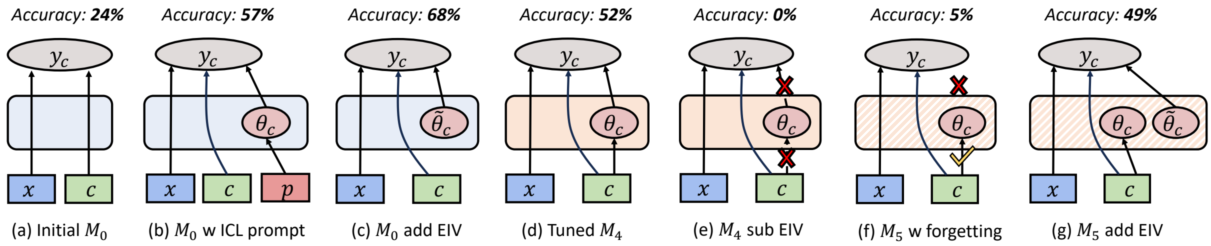


Figure 4: Illustration of the instruction vector hypothesis. Here, x represents the context, c stands for a specific instruction, y_c is the desirable output, and θ_c denotes the instruction vector. From (a) to (g), it visually details how these variables interact under different model conditions, with the accuracy above correlating to the respective performance on the CommonsenseQA task. The model configuration depicted in (d) is identified as the best state.

demonstrated in Fig. 4 b,c. The detailed procedure is outlined below:

First, we start by gathering the task-conditioned activation for each model head by averaging the ICL input representation of the given task D^c, i.e.,

$$\bar{h}_{lj}^{c} = \frac{1}{|D^c|} \sum_{(x_i,c)\in D^c} h_{\ell j}\left([p_i, x_i, c]\right). \tag{1}$$

Where $p_i = [(x_1, c, y_1^c), ..., (x_N, c, y_N^c)]$ represents the N-shot ICL prompt text made up of held-out samples of task c, h_{lj} is the model activation at the last token, layer l and position j, and $\bar{h}_{lj}^c$ represents the task-conditioned activations.

Then to assess the existence of a cause-and-effect relationship between $\bar{h}_{lj}^c$ and correct output, we employ causal mediation analysis. The model will run on a counterfactual ICL input $[\hat{p}_i, x_i, c]$ incorporating a label-shuffled prompt $\hat{p}_i = [(x_1, c, \hat{y}_1^c), ..., (x_N, c, \hat{y}_N^c)]$, typically leading to incorrect outcomes. We then substitute the value of the specific head with the task-specific conditioned activation $\bar{h}_{lj}^c$ and calculate its causal effect (CE) on the model's output.

$$\begin{aligned}\mathrm{CE}_{lj}([\hat{p}_i, x_i, c]) = P(y_i^c \mid [\hat{p}_i, x_i, c], M_{h_{lj}^c \to \bar{h}_{lj}^c}) \\ - P(y_i^c \mid [\hat{p}_i, x_i, c], M).\end{aligned} \tag{2}$$

Here, $M_{h_{lj}^c \to \bar{h}_{lj}^c}$ denotes the model with a replacement operation on attention head (l, j) at last token of the input sentence. A higher CE suggests that the specific head's state is crucial in enabling accurate predictions, denoting the encoding of more task-relevant information. For each head at layer l and position j,we adopt the approach proposed by Todd et al. (2023) to calculate the average CE across a variety of tasks. Subsequently, we identify the top 10 heads with the highest average CE (recorded as set $\mathcal{S}$) as the most critical in conveying task-relevant information. The task vector θ_c is is then obtained by aggregating the task-conditioned activation from the attention heads in the set $\mathcal{S}$, i.e., $\theta_c = \sum_{a_{\ell j}\in\mathcal{S}} \bar{h}_{lj}^c$.

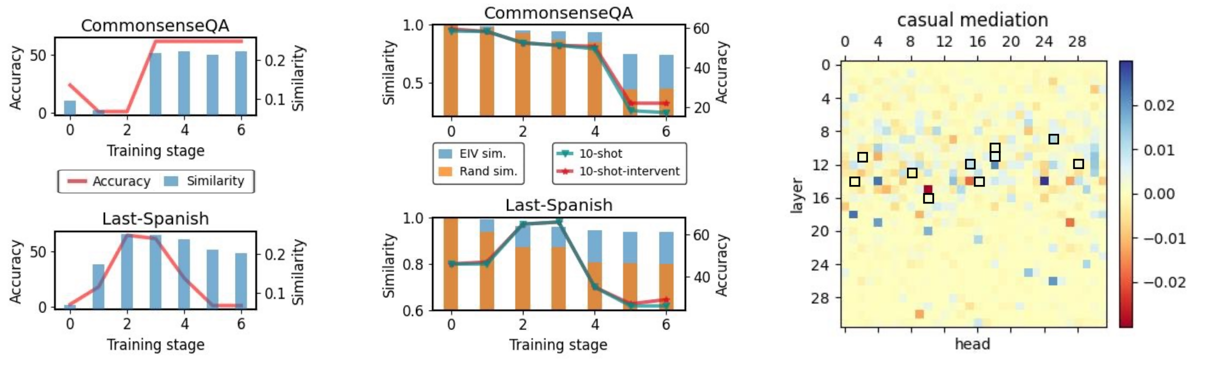
We then evaluates the effectiveness of the Instruction Vector (θ_c) through intervention experiments on the initial model across multiple datasets. The detail experiments can be found in Appendix E. Results show that the IV significantly influences the output behavior for specific tasks, with its introduction notably improving zero-shot performance in certain tasks and removal diminishing the model's ability to produce correct outputs. This suggests that the model's specific abilities can be identified and analyzed by studying the corresponding IV.

3.3 Fine-tuning Dynamics

In this series of experiments, we aim to explore how the Instruction Vector (IV) evolves during continual instruction tuning to better understand the mechanisms underlying forgetting.

Finding 1. *Alignment between the fine-tuned computation graph and the IV-associated computation graph correlates with task performance.* Fig. 5 shows the relationship between zero-shot performance and the similarity of hidden states to their respective instruction vector, measuring alignment through the cosine similarity $\mathrm{Cosine}(h_l, \theta_c)$. This similarity is utilized to reflects the alignment between the computation graphs, with h_l denotes the hidden state of the l-th layer. The maximum value across all layers is reported.

Post fine-tuning, the model appears to incorporate θ_c into the hidden states, evidenced by a similarity score of 0.249 for Last-Spanish in stage 2, correlating with improved task accuracy (65%). Conversely, a performance decline is linked to a decrease in similarity. For instance, in the Last-Spanish task, accuracy fell from 65% to 1% in stage 3-6, alongside a drop in similarity. On the other



(a) (b) (c)

Figure 5: **(a)**: Relationship between zero-shot task performance (red line) and similarity (blue bar) between hidden states and IV during tuning on FUNC benchmark. Here, CommonsenseQA is the general evaluation set and Last-Spanish is the second training task. **(b)**: Representation shift on IV/Random position with 10-shot performance during tuning. **(c)**: Casual mediate analysis results on model fine-tuned after 6-th stage on TRACE benchmark. The report value is casual effect and black boxes denote the top-10 heads of the initial model.

hand, in CommonsenseQA, consistent similarity coincided with stable performance, underscoring the importance of maintaining the IV-associated computation graph for task effectiveness.

Finding 2. *The consistency of IV before and after fine-tuning does not play a key role in preventing forgetting.* Fig. 5 (b) shows shifts in the instruction vector θ_c and representation from random positions during training. The results of two test datasets that exhibit significant forgetting are reported in the figure, including the CommonsenseQA in TRACE and Last-Spanish in FUNC. "IV sim." in the diagram refers to $\text{Cosine}(\theta_c^0, \theta_c^i)$, where θ_c^i is the IV after fine-tuning the i-th task. "Rand sim." tracks changes from 10 randomly chosen head outputs, averaged over 100 seeds. Despite IV maintaining stability at 0.95/0.79 even into the 6-th phase, compared to random similarity scores of 0.8/0.48, significant model forgetting still occurs by the 6-th phase, with accuracy for Last-Spanish falling to 26% and CommonsenseQA to 17.25%.

Furthermore, experiments with IV-related interventions, where hidden states contribute to IV in the fine-tuned model were replaced with their initial values (stage 0), are shown by the red line in the Fig. 5 (b). The purpose of this experiment was to re-activate the model's capacity to handle the specific task by fully recovering the representation of IV. However, results suggested minimal effectiveness. The findings indicate that after training, the model cannot implicitly utilize θ_c; hence, the output y becomes detached from θ_c, disrupting the computation graph. Thus, changes in IV before and after fine-tuning do not contribute to the observed forgetting.

Finding 3. *Model forgetting stems from suppression by new specialized patterns.* We conducted a causal mediate analysis (Sec. 3.2) on the fine-tuned model and observed a significant shift in the set $\mathcal{S}$ of casual attention heads. The results are reported in Fig. 5 (c). This suggests that the original capability of the model to process tasks was suppressed by new, specialized patterns, leading to a decrease in general capability.

Furthermore, we conducted an intervention experiment on the CommonsenseQA task with the model fine-tuned on the TRACE benchmark (refer to Fig. 7). The results show that the model exhibited significant forgetting in both 0-shot and 10-shot performance, dropping to 0.03 and 0.15, respectively. However, integrating IV into the model (as shown in Fig. 1(g)), i.e., $h_l = h_l + \theta_c$, result in a substantial recovery in model performance, achieving 0.47 with the current model's IV and 0.49 with the initial model's IV. This demonstrates that by explicitly adding IV back to the computation graph, the model can still adhere to current task instructions, indicating that the observed forgetting is not due to a loss of the model's ability to handle instructions.

In conclusion, our analysis suggests that forgetting in large language models (LLMs) results from a dynamic conflict between the dominance and suppression of existing computation graphs and new, specialized reasoning patterns learned from fine-tuning. This extends previous findings Kotha et al. (2024) by utilizing IV framework to explore the underlying processes of forgetting in these models and confirming its theoretical underpinnings.

4 Refinement of Training Methods to Mitigate Forgetting in LLMs

Our previous research highlighted the critical role of the Instruction Vector (IV)-associated computation graph in Large Language Models (LLMs), crucial for maintaining the model's original capabilities. This insight prompted a reassessment of the training approaches to minimize forgetting. In this section, we show that fine-tuning guided by the *Instruction Vector* helps balance the model's existing capabilities with new learning. This led us to reevaluate our training methods to prevent forgetting. This method, combined with existing continual learning algorithms, effectively reduces the forgetting of general abilities while preserving in-context reasoning capabilities, with minimal impact on plasticity.

Instruction vector guided fine-tuning. In our analysis, we established a direct link between the IV-associated computation graph and the model's inherent task processing abilities. Forgetting typically occurs when the model's output becomes independent of the computation graph post-tuning. To address this, we propose an IV-guided training mechanism aimed at preserving capabilities before and after fine-tuning:

Initially, to utilize of the capabilities introduced by the IV, we propose a progressive IV intervention training. At training's start, the IV is explicitly included, with its influence gradually diminishing from 1 to 0 as training advances. This inclusion helps the model adhere to the computation graph outlined earlier, thus mitigating the overshadowing of existing capabilities by new learning. The original training objective is reformulated as:

$$\min_{M} \frac{1}{N}\sum_{i=1}^{N} \ell\left(y_i, M_{h^c_{lj} \to h^c_{lj} + s * \bar{h}^c_{lj}}(c, x_i)\right), \quad (3)$$

where $M_{h^c_{lj} \to h^c_{lj} + s * \bar{h}^c_{lj}}$ denotes the intervention model on the causal attention heads set i.e., $(l, j) \in \mathcal{S}$. s is a scaling factor that gradually decreases from 1 to 0 during training.

Furthermore, we introduce an IV-based KL-divergence loss function to better align the behaviour of fine-tuned computation graph with the IV indications:

$$\ell_{KL} = -KL[P(y^c|[c,x], M) \| P(y^c|[c,x], M_{h^c_{lj} \to h^c_{lj} + \bar{h}^c_{lj}})]. \quad (4)$$

This IV-guided fine-tuning approach leverages the existing knowledge within the model to direct the fine-tuning process, ensuring that the model retains a robust computation graph after fine-tuning and minimizes the impact of newly introduced knowledge on past knowledge and abilities.

Experimental Setup. Following the continual instruction tuning setup in Sec. 2, we test our newly proposed method on TRACE and FUNC benchmarks additionally with a LONG sequence continual learning benchmark (Razdaibiedina et al., 2023) with 15 tasks. For the held-out evaluation set, we utilize Hellaswag, ARC-challenge, CommonsenseQA, and MMLU-social. The experiments were conducted on the Llama2-7B-chat model, demonstrating its effectiveness in combination with existing continual learning methods, such as incremental Lora (Hu et al., 2021) (**IncLora**), Learning without forgetting (Li and Hoiem, 2017) (**Lwf**), Elastic weight consolidation (Kirkpatrick et al., 2017) (**Ewc**), Orthogonal Lora (Wang et al., 2023a) (**OLora**). In our comparison, we prioritized training with hyper-parameters mentioned in previous works. We loaded the base LM into torch.bfloat16 to save memory and ran the experiments on 4 NVIDIA A100 GPUs.

To evaluate the performance of proposed algorithms, we utilize the average zero-shot held-out performance $HP = \frac{1}{n}\sum_{i=1}^{n} a_T^{h_i}$ to measure shift in general capabilities, average in-content held-out performance $IP = \frac{1}{n}\sum_{i=1}^{n} \hat{a}_T^{h_i}$ to evaluate forgetting in reasoning abilities, and overall training performance $OP = \frac{1}{T}\sum_{i=1}^{T} a_T^{t_i}$ to assess the degree of catastrophic forgetting on newly learned abilities. Here, a_j^i represents the zero-shot evaluation score on the evaluation task i after sequentially learning the j-th task. $\hat{a}$ denotes the in-context evaluation score. h_i and t_i denotes the i-th held-out evaluation set and i-th training task, respectively.

Results. Table 1 shows the continual instruction tuning performance on three benchmarks, leading to several key observations:

Observation 1: IV-guided training significantly prevents the loss of general and reasoning capabilities. Unlike most continual learning methods, which struggle with substantial forgetting of general abilities, our IV-guided training effectively mitigates this issue, resulting in an average forgetting rate on HP of -0.16, compared to 5.03. Additionally, it enhances in-context performance from 37.90 to 50.05, underscoring the benefits of maintaining

Method	TRACE			LONG			FUNC		
	HP	IP	OP	HP	IP	OP	HP	IP	OP
Init	52.76	54.31	18.68	52.76	54.31	42.62	52.76	54.31	11.70
IncLora	48.69	26.73	47.60	50.28	49.75	78.11	53.12	51.78	43.34
+ IVG	54.75 (**+6.06**)	45.85 (**+19.1**)	47.20	52.54 (**+2.26**)	51.64 (**+1.89**)	77.41	54.36 (**+1.24**)	53.89 (**+2.11**)	69.48
Ewc	52.80	43.96	47.70	45.83	43.61	73.62	52.05	50.33	38.46
+ IVG	54.94 (**+2.14**)	54.58 (**+10.6**)	46.69	52.38 (**+6.55**)	53.36 (**+9.75**)	71.71	54.22 (**+2.17**)	54.03 (**+3.70**)	38.56
Lwf	52.71	54.44	34.68	51.73	52.40	69.39	53.33	54.43	57.91
+ IVG	52.93 (**+0.22**)	54.49 (**+0.05**)	34.65	53.85 (**+0.56**)	53.89 (**-0.40**)	70.60	53.59 (**+0.26**)	54.23 (**-0.20**)	61.92
OLora	36.68	26.48	38.22	50.07	45.87	77.68	54.13	52.38	42.12
+ IVG	49.08 (**+12.4**)	46.35 (**+19.9**)	39.78	52.05 (**+1.98**)	51.48 (**+5.61**)	76.98	53.94 (**-0.19**)	53.90 (**+1.52**)	58.13

Table 1: Performance of baseline and their improved version with Instruction Vector Guided (**IVG**) training on three benchmarks (all results reported in this paper are averaged over 4 random seeds).

the computation graph.

Observation 2: IV-guided training does not compromise the plasticity in learning new tasks. This approach shows only a slight reduction in the OP metric, with changes of -0.03 and -0.55 for TRACE and LONG, respectively. This is in sharp contrast to the Lwf algorithm, which significantly reduces adaptability, resulting in a dramatic 12.92 drop in OP on TRACE compared to IncLora.

Observation 3: The likelihood of forgetting general abilities increases with the complexity of learning tasks. The benchmarks in Table 1, ranked from simplest to most complex—FUNC, LONG, TRACE—show escalating HP forgetting rates from -0.40 to 2.89 and then to 5.04. The IV-guided training method effectively manages tasks across varying complexities, demonstrating its robustness in handling different learning challenges.

5 Related work

Catastrophic forgetting in fine-tuned language models. Fine-tuning foundational LLMs (Touvron et al., 2023a,b) has become a generic technique for enhancing their capacity of following instructions (Wei et al., 2022; Zhang et al., 2024a,b) and mastering domain-specific content (Yue et al., 2023; Christophe et al., 2024). However, adopting such technique can have a negative effect of hurting the original ability of LLMs, which is widely known as Catastrophic Forgetting (Kirkpatrick et al., 2017; Zhai et al., 2023; Luo et al., 2024; Kotha et al., 2024; Wu et al., 2024b). In context of LLMs, existing approaches towards mitigating this issue can mostly be categorized into three types: regularizing the update of model parameters (Kirkpatrick et al., 2017; Huang et al., 2021; Cha et al., 2021), replaying previous or self-synthesized data (Scialom et al., 2022; Huang et al., 2024a) and resisting interference via parameter-efficient fine-tuning (Razdaibiedina et al., 2023; Wang et al., 2023a).

Mechanistic analysis to fine-tuning. Existing works on analyzing the internal mechanism (Räuker et al., 2023; Ferrando et al., 2024) of fine-tuning mainly focus on the question that how LLMs acquire new capacity in the learning process, arguing that models learn a minimal transformation on top of the original capability (Jain et al., 2024) (wrappers), subtractable and reusable parameter shift vectors (Huang et al., 2024b; Gao et al., 2024) (task vectors) and to align input queries with their internal knowledge that are already acquired in the pre-training stage (Ren et al., 2024). Nevertheless the inherent reason for the forgetting issue brought by fine-tuning currently remains unclear, and hence our work instead targets on this important point.

6 Conclusion

In our study, we introduce Instruction Vector (IV), which enables detailed analysis of LLMs task processing capabilities. By analyzing IV dynamics before and after training, we show that forgetting is caused by the overlay of new reasoning patterns over pre-existing skills, while the performance can be recovered by adding the IV to the computation graph. Additionally, our proposal of IV-guided training as a fine-tuning method successfully reduces forgetting by maintaining harmony between the model's computation graph and the IV-associated one. These findings offer valuable insights into the internal mechanisms causing forgetting in LLMs and are expected to contribute to advancing the development and application of LLMs alignment.

7 Limitation

The IV-guided training method does not directly address the problem of forgetting newly learned knowledge in most cases, and needs to be combined with existing continual learning methods to acquire this ability. This is because we overcome forgetting by preserving the computation graph, which indicates the existing capabilities, making it unable to protect newly acquired knowledge. Interestingly, in the FUNC dataset, our method significantly reduced forgetting of new knowledge on IncLora and OLora. These tasks have simple and deterministic instructions, which may allow the model to integrate new capabilities with the constructed computation graph during IV-guided training, thus overcoming forgetting. This inspires us to investigate the adaptability and generalization of the computation graph in future research for more refined learning of new knowledge.

Second, we aggregate attention heads to extract the Instruction vector in this paper. Although this method is fast and efficient, it is susceptible to input noise and may suffer from insufficient expressiveness. Therefore, we plan to use optimization-based methods in future to extract a more generalized and accurate Instruction vector.

Finally, due to limitations in experimental resources, we did not conduct experiments on multiple backbones. In the future, we will validate our hypothesis about forgetting on more LLMs.

References

Antoine Bordes, Y-Lan Boureau, and Jason Weston. 2016. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Sungmin Cha, Hsiang Hsu, Taebaek Hwang, Flavio Calmon, and Taesup Moon. 2021. {CPR}: Classifier-projection regularization for continual learning. In *International Conference on Learning Representations*.

Lingjiao Chen, Matei Zaharia, and James Zou. 2023. How is chatgpt's behavior changing over time? *arXiv preprint arXiv:2307.09009*.

Clément Christophe, Praveen K Kanithi, Prateek Munjal, Tathagata Raha, Nasir Hayat, Ronnie Rajan, Ahmed Al-Mahrooqi, Avani Gupta, Muhammad Umar Salman, Gurpreet Gosal, Bhargav Kanakiya, Charles Chen, Natalia Vassilieva, Boulbaba Ben Amor, Marco AF Pimentel, and Shadab Khan. 2024. Med42 – evaluating fine-tuning strategies for medical llms: Full-parameter vs. parameter-efficient approaches. *Preprint*, arXiv:2404.14779.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, et al. 2023. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*.

Javier Ferrando, Gabriele Sarti, Arianna Bisazza, and Marta R. Costa-jussà. 2024. A primer on the inner workings of transformer-based language models. *Preprint*, arXiv:2405.00208.

Lei Gao, Yue Niu, Tingting Tang, Salman Avestimehr, and Murali Annavaram. 2024. Ethos: Rectifying language models in orthogonal parameter space. *Preprint*, arXiv:2403.08994.

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. *arXiv preprint arXiv:2310.15916*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

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. *arXiv preprint arXiv:2106.09685*.

Jianheng Huang, Leyang Cui, Ante Wang, Chengyi Yang, Xinting Liao, Linfeng Song, Junfeng Yao, and Jinsong Su. 2024a. Mitigating catastrophic forgetting in large language models with self-synthesized rehearsal. *Preprint*, arXiv:2403.01244.

Shih-Cheng Huang, Pin-Zu Li, Yu-Chi Hsu, Kuang-Ming Chen, Yu Tung Lin, Shih-Kai Hsiao, Richard Tzong-Han Tsai, and Hung yi Lee. 2024b. Chat vector: A simple approach to equip llms with instruction following and model alignment in new languages. *Preprint*, arXiv:2310.04799.

Yufan Huang, Yanzhe Zhang, Jiaao Chen, Xuezhi Wang, and Diyi Yang. 2021. Continual learning for text classification with information disentanglement based regularization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2736–2746, Online. Association for Computational Linguistics.

Samyak Jain, Robert Kirk, Ekdeep Singh Lubana, Robert P. Dick, Hidenori Tanaka, Tim Rocktäschel, Edward Grefenstette, and David Krueger. 2024. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks. In *The Twelfth International Conference on Learning Representations*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Suhas Kotha, Jacob Mitchell Springer, and Aditi Raghunathan. 2024. Understanding catastrophic forgetting in language models via implicit inference. In *The Twelfth International Conference on Learning Representations*.

Zhaoyi Li, Gangwei Jiang, Hong Xie, Linqi Song, Defu Lian, and Ying Wei. 2024. Understanding and patching compositional reasoning in llms. *arXiv preprint arXiv:2402.14328*.

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947.

Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2024. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *Preprint*, arXiv:2308.08747.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Judea Pearl. 2013. Interpretation and identification of causal mediation. *ERN: Other Econometrics: Econometric Model Construction*.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Anastasia Razdaibiedina, Yuning Mao, Rui Hou, Madian Khabsa, Mike Lewis, and Amjad Almahairi. 2023. Progressive prompts: Continual learning for language models. In *The Eleventh International Conference on Learning Representations*.

Mengjie Ren, Boxi Cao, Hongyu Lin, Cao Liu, Xianpei Han, Ke Zeng, Guanglu Wan, Xunliang Cai, and Le Sun. 2024. Learning or self-aligning? rethinking instruction fine-tuning. *Preprint*, arXiv:2402.18243.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Tal Remez, Jérémy Rapin, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Tilman Räuker, Anson Ho, Stephen Casper, and Dylan Hadfield-Menell. 2023. Toward transparent ai: A survey on interpreting the inner structures of deep neural networks. *Preprint*, arXiv:2207.13243.

Thomas Scialom, Tuhin Chakrabarty, and Smaranda Muresan. 2022. Fine-tuned language models are continual learners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6107–6122, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. 2023. Large language models in medicine. *Nature medicine*, 29(8):1930–1940.

Eric Todd, Millicent L Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2023. Function vectors in large language models. *arXiv preprint arXiv:2310.15213*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems*, volume 33, pages 12388–12401. Curran Associates, Inc.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. 2023a. Orthogonal subspace learning for language model continual learning. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Xiao Wang, Yuansen Zhang, Tianze Chen, Songyang Gao, Senjie Jin, Xianjun Yang, Zhiheng Xi, Rui Zheng, Yicheng Zou, Tao Gui, et al. 2023b. Trace: A comprehensive benchmark for continual learning in large language models. *arXiv preprint arXiv:2310.06762*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Chengyue Wu, Yukang Gan, Yixiao Ge, Zeyu Lu, Jiahao Wang, Ye Feng, Ping Luo, and Ying Shan. 2024a. Llama pro: Progressive llama with block expansion. *arXiv preprint arXiv:2401.02415*.

Tongtong Wu, Linhao Luo, Yuan-Fang Li, Shirui Pan, Thuy-Trang Vu, and Gholamreza Haffari. 2024b. Continual learning for large language models: A survey. *Preprint*, arXiv:2402.01364.

Shengbin Yue, Wei Chen, Siyuan Wang, Bingxuan Li, Chenchen Shen, Shujun Liu, Yuxuan Zhou, Yao Xiao, Song Yun, Xuanjing Huang, and Zhongyu Wei. 2023. Disc-lawllm: Fine-tuning large language models for intelligent legal services. *Preprint*, arXiv:2309.11325.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.

Yuexiang Zhai, Shengbang Tong, Xiao Li, Mu Cai, Qing Qu, Yong Jae Lee, and Yi Ma. 2023. Investigating the catastrophic forgetting in multimodal large language models. *Preprint*, arXiv:2309.10313.

Renrui Zhang, Jiaming Han, Chris Liu, Aojun Zhou, Pan Lu, Yu Qiao, Hongsheng Li, and Peng Gao. 2024a. LLaMA-adapter: Efficient fine-tuning of large language models with zero-initialized attention. In *The Twelfth International Conference on Learning Representations*.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. 2024b. Instruction tuning for large language models: A survey. *Preprint*, arXiv:2308.10792.

A Datasets

Three continual instruction tuning benchmarks and severel general evaluation datasets are adopts in this paper. The detailed information is as follows:

TRACE benchmark. TRACE benchmark is released by Wang et al. (2023b) for the study of forgetting in LLMs, which consists of 8 different complex generation tasks including multi-choice QA, code generation, mathematical reasoning and summary. Without loss of generaliztion, we select 6 out of 8 raw tasks to construct the training sequence as our experiments setup. The statistical information is listed in Table 2, while order in Table 6

The training epoch for this benchmark is 5 for C-STANCE, Py150, NumGLUE-cm, 3 for FOMC and ScienceQA, and 7 for MeetingBank. We evaluate them with a self-construct evaluation code based on OpenCompass code framework.

LONG benchmark. LONG benchmark is widely utilized in existing continual learning works Wang et al. (2023a); Razdaibiedina et al. (2023) with 15 task. The training epoch is set to 1 for each task following (Wang et al., 2023a). The statistical information is listed in Table 4.

FUNC benchmark. FUNC benchmark is adapted from the datasets in Todd et al. (2023), in which tasks have clear and simple instructions. For example, task Verb-Spanish and Last-Spanish are both translation task but differ in the selection from list. The training epoch is set to 10 for each task. The statistical information is listed in Table 4.

General evaluation sets. For the general evaluation datasets, we utilize Hellaswag (Zellers et al., 2019), ARC-challenge (Clark et al., 2018), CommonsenseQA (Talmor et al., 2018), and MMLU-social (Hendrycks et al., 2020). The datasets is downloaded from `https://github.com/open-compass/opencompass` and evaluate with the OpenCompass code framework.

B Input template

In this paper, the instruction template is divided into two parts, refer to Sec. 2. The first part corresponds to knowledge probability, namely (x, y), and the second part corresponds to Instruction probability, namely (x, c, y^c). The specific template content used for each dataset is given below, as show in Table 5, Table 7, and Table 8.

Dataset	Source	Category	Avg len	Metric	Language	#data
ScienceQA	Science	Multi-Choice QA	210	Accuracy	English	5,000
FOMC	Finance	Multi-Choice QA	51	Accuracy	English	5,000
MeetingBank	Meeting	Summary	2853	ROUGE-L	English	5,000
C-STANCE	Social media	Multi-Choice QA	127	Accuracy	Chinese	5,000
Py150	Github	Code generation	422	Edim similarity	Python	5,000
NumGLUE-cm	Math	Math reasoning	32	Accuracy	English	5,000

Table 2: A summary of dataset statistics in TRACE includes information on the source of the context, average length in terms of word count for English, German, and code datasets, and character count for Chinese.

Dataset	Source	Category	Avg len	Metric	Language	#data
Yelp	Yelp reviews	Sentiment analysis	757	Accuracy	English	5,000
SST2	Movie reviews	Sentiment analysis	62	Accuracy	English	2,000
Amazon	Amazon reviews	Sentiment analysis	458	Accuracy	English	5,000
IMDB	Movie reviews	Sentiment analysis	1,340	Accuracy	English	2,000
DBpedia	Wikipedia	Topic classification	324	Accuracy	English	14,000
Yahoo	Yahoo Q&A	Topic classification	562	Accuracy	English	10,000
AG News	News	Topic classification	259	Accuracy	English	4,000
WiC	Lexical database	Disambiguation	93	Accuracy	English	2,000
QQP	Quora	Paraphrase	158	Accuracy	English	2,000
RTE	News, Wikipedia	NLI	365	Accuracy	English	2,000
MNLI	Multi	NLI	205	Accuracy	English	3,000
CB	Multi	NLI	365	Accuracy	English	250
COPA	blogs, encyclopedia	Question answering	161	Accuracy	English	400
BoolQ	Wikipedia	Question answering	655	Accuracy	English	2,000
MultiRC	SuperGLUE	Question answering	1728	Accuracy	English	2,000

Table 3: A summary of dataset statistics in LONG.

Dataset	Source	Category	Avg len	Metric	Language	#data
Alphabetically_last_of_5	–	Extractive, Capital	144	Accuracy	English	700
Choose_last_of_5_spanish	–	Extractive, Translation	109	Accuracy	English, Spanish	700
AG News	News	Topic classification,QA	285	Accuracy	English	1,500
Object_v_concept_5_spanish	–	Extractive, Translation	106	Accuracy	English, Spanish	700
Verb_v_adjective_5_spanish	–	Extractive, Translation	106	Accuracy	English, Spanish	700
Sentiment	–	Sentiment analysis,QA	75	Accuracy	English	816

Table 4: A summary of dataset statistics in FUNC.

Task	Template
Yelp, SST2, Amazon, IMDB	"Input": "What is the sentiment of the following paragraph? [x] Choose one from the option.", "Output": "[y]"
DBPedia, Yahoo, AG NEws	"Input": "What is the topic of the following paragraph? [x] Choose one from the option.", "Output": "[y]"
QQP	"Input": "Whether the [x_1] and the [x_2] have the same meaning? Choose one from the option.", "Output": "[y]"
RTE, MNLI, CB	"Input": "What is the logical relationship between the [x_1] and the [x_2]? Choose one from the option.", "Output": "[y]"
BoolQA	"Input": "According to the following passage, is the question true or false? [x] Choose one from the option.", "Output": "[y]"
MultiRC	"Input": "According to the following passage and question, is the candidate answer true or false? [x] Choose one from the option.", "Output": "[y]"
WiC	"Input": "Given a word and two sentences, whether the word is used with the same sense in both sentence? Choose one from the option.", "Output": "[y]"

Table 5: Input template for tasks in LONG benchmark.

Benchmark	Task Sequence
TRACE	C-STANCE→ FOMC→ MeetingBank→ Py150→ ScienceQA → NumGLUE-cm
LONG	Yelp→ Amazon→ MNLI→ CB→ COPA→ QQP→ RTE→ IMDB→ SST2→ DBpedia→ AG News→ Yahoo→ MultiRC→ BoolQ→ WiC
FUNC	Verb-Spanish → Last-Spanish→ Sentiment-mc→ Object-Spanish→ Alphabetically-Capital → AGNews-mc

Table 6: The orders used for each benchmark.

Task	Prompts
ScienceQA	"Input": $[x]$ "Output": $[y]$
FOMC	"Input": "Text: $[x]$ The monetary policy stance of above text is ", "Output": "$[y]$"
C-STANCE	(Translate Chinese to English) "Input": "Text: $[x_1]$ Object: $[x_2]$ The attitude of above text towards object is", "Output": "$[y^c]$"
Last-Spanish	"Input": "Choose the last item in the list. $[x]$", "Output": "$[y^c]$"
Object-Spanish	"Input": "Choose the object in the list. $[x]$", "Output": "$[y]$"
Verb-Spanish	"Input": "Choose the verb in the list. $[x]$", "Output": "$[y]$"
Alphabetically-Capital	"Input": "Choose the last item in the order of alphabetically in the list. $[x]$", "Output": "$[y]$"
AGNews-mc	"Input": "Classify the following news with the label Business, Science, Sports, and World. $[x]$ ", "Output": "$[y]$"
Sentiment-mc	"Input": "$[x]$", "Output": "$[y]$"

Table 7: Input template for calculating knowledge probability for different tasks.

Task	Prompts
ScienceQA	"Input": "Choose an answer for the following question and give your reasons. Question: $[x]$ Answer:", "Output": "$[y^c]$"
FOMC	"Input": "What is the monetary policy stance for the following text? A. dovish, B. hawkish, C. neutral. Choose one from A, B and C. Text: $[x]$ Stance:", "Output": "$[y^c]$"
C-STANCE	(Translate Chinese to English) "Input": "Determine the attitude of the following text towards the specified object. Select one: A. Support, B. Oppose, C. Neutral. Output A, B or C. Text: $[x_1]$ Object: $[x_2]$ Attitude:", "Output": "$[y^c]$"
MeetingBank	"Input": "Write a summary of the following meeting transcripts. Meeting transcripts: $[x]$ Summary:", "Output": "$[y]$"
Py150	"Input": "<s> [x]", "Output": "$[y]$"
NumGLUE-cm	"Input": "Solve the following math problem. Question: $[x]$ Answer:", "Output": "$[y]$"
Last-Spanish	"Input": "Choose the last item in the list and translate to spanish. $[x]$", "Output": "$[y^c]$"
Object-Spanish	"Input": "Choose the object in the list and translate to spanish. $[x]$", "Output": "$[y^c]$"
Verb-Spanish	"Input": "Choose the verb in the list and translate to spanish. $[x]$", "Output": "$[y^c]$"
Alphabetically-Capital	"Input": "Choose the last item in the order of alphabetically in the list and print in the capital form. $[x]$", "Output": "$[y^c]$"
AGNews-mc	"Input": "$[x]$ A: Business B: Science C: Sports D: World", "Output": "$[y^c]$"
Sentiment-mc	"Input": "$[x]$ a: positive b: negative", "Output": "$[y^c]$"

Table 8: Input template for calculating instruction probability and training for different tasks.

C Implementation

We adopt LLAMA2-7B-Chat (Touvron et al., 2023b) as the base model, with its effectiveness in both understanding world knowledge and following instructions. Without specific notification, the model is fine-tuned with LORA approach (Hu et al., 2021), where the rank dimension set to 8 and the target module is query and value weight matrices. For IncLora, OLora, and Lwf methods, a new adapter is initialized at the beginning of learning new task while keep the previous Lora adapters fixed. For Ewc, only one big adapter is initialized during the sequential learning, where rank is set to 48 for TRACE and FUNC, and 60 for LONG.

The maximum input sequence length is set to 512 and the maximum output sequence length is set to 50. We train the model with the decoder only task calculating gradient only on the output tokens. We use an Adam optimizer with a weight decay of 0.01 and the learning rate set to 1e-4 for TRACE and FUNC, 1e-3 for LONG (following (Wang et al., 2023b)). The batch size is set to 8 and accumulate gradient step is set to 2 for each GPU while we run on 4 A100 GPUs with Deepspeed. The training size and epochs can be found in the introduction of datasets.

As for the hyperparameters, we perform a grid search on the scale of KL-divergence loss within [1, 0.5, 0.25, 0.05, 0.01] and set 0.05 as the final choice. For the hyperparameters of existing continual learning methods, I refer to the well-searched value reported in previous paper.

D Implementation Detail of Instruction Vector Framework

When extracting the Instruction Vector from in-context samples, we use 10-shot input prompt randomly selected from held-out training dataset. The task-conditioned activations are average on samples filtered with correct 10-shot answer from the validation set with 200 samples. As for the set $\mathcal{S}$ of the casual attention heads, we follow the position in Todd et al. (2023) and validate its efficiency on our own datasets. Specifically, the set $\mathcal{S}$ is $[(14, 1), (11, 2), (9, 25), (12, 15), (12, 28), (13, 7), (11, 18), (12, 18), (16, 10), (14, 16)]$.

E Effectiveness of Instruction Vector

To assess the effectiveness of the extracted θ_c, referred to as the Instruction Vector (IV) in this study, we conduct a series of intervention experiments across multiple datasets (see Fig. 6) on the initial model. These experiments consisted of either inserting or removing an IV at the hidden states of a specific layer at the the last token position, to examine the influence on the model output. More precisely, in the transformer's forward residual stream, the instruction vector θ_c modifies the hidden states at a select layer l as $h_l = h_l + \theta_c$.

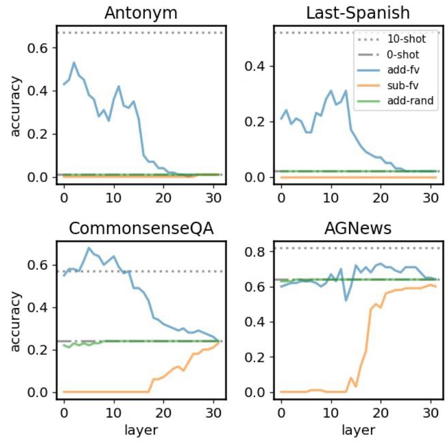


Figure 6: Intervention results on four datasets via Enhanced Instruction Vector.

We reported the intervention findings on four distinct datasets: 1) CommensenseQA, multiple-choice questions on common sense reasoning; 2) Antonym, a task aimed at generating antonyms; 3) AGNews, a text classification task with the article's category as the label; and 4) Last-Spanish, a task that output the Spanish translation of the list's final item. The results highlighted that the IV directly affects the model's output behavior for specific tasks. In tasks such as Antonym, Last-Spanish, and CommonsenseQA, introducing IV significantly improved the zero-shot performance from a low level. Conversely, in the cases of AGNews and CommonsenseQA, removing the IV resulted in a deterioration of the model's ability to produce the correct output. In contrast, interventions with random vectors had a negligible effect on the model. These findings indicate that the specific capabilities of the model can be identified and analyzed by examining the dynamics of the corresponding IV.

F Recovery with Instruction Vector

We conducted an intervention experiment on the CommonsenseQA task with the fine-tuned model on the TRACE benchmark (refer to Fig. 7). The results show that the model exhibited significant forgetting in both 0-shot and 10-shot performance, dropping to 0.03 and 0.15, respectively. However, integrating IV into the model (as shown in Fig. 1(g)), i.e., $h_l = h_l + \theta_c$, resulted in a substantial recovery in model performance. Performance reached 0.47 when using IV derived from the current model and 0.49 with IV from the initial model.

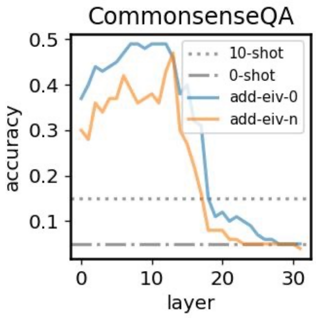


Figure 7: The intervention results on model sequentially fine-tuned on TRACE benchmark.