Knowledge Editing

Large Language Models: Introduction and Recent Advances

ELL881 · AIL821

Tanmoy Chakraborty
Associate Professor, IIT Delhi
https://tanmoychak.com/

Mistral AI announces Pixtral !

Announced on September 17, 2024

Mistral AI Blog

Pixtral 12B is the first multimodal model by Mistral AI.

Pixtral consists of a **new 400M parameter vision encoder trained from scratch**, and a **12B parameter multimodal decoder based on Mistral Nemo**. Pixtral supports multiple images in the long context window of 128k tokens.

Pixtral supports variable image sizes and aspect ratios. It understands both natural images and documents, **achieving 52.5% on the MMMU reasoning benchmark, surpassing a number of larger models**

LLMs Need Regular Updates

https://arxiv.org/pdf/2310.16218

Issues with Finetuning

- Computationally expensive

  LLaMA was trained for **21 days** on **2,048 A100 GPUs**, costing over **$2.4M** and emitting over **1,000 tons** of CO2 [Hartvigsen et al., NeurIPS'23]

- Unconstrained editing

  Fine-tuning LLMs alters the pre-trained parameters without constraints, leading to the overfitting problem [Wortsman et al, CVPR'22]

Knowledge Editing: Definition

Knowledge Edit: A factual information can be presented as a triplet < s, r, o >

where,
s : subject r : relation o : object

Example: **“The Space Needle** is **located** in the city of **Seattle**’

s : subject : The Space Needle
r : relation : Location
o : object : Seattle

It can be expressed as <The space Needle, Location, Seattle>

Goal

- Update the objective from ‘o’ to ‘o*’ i.e. < s, r, o > to < s, r, o* >

  Example:

  < The space Needle, Location, Seattle > **->** <The space Needle, Location, Goa >
  < The US, President, Barack Obama > **->** < The US, President, Joe Biden >
  < Earth, Highest point, Mount Everest > **->** < Earth, Highest point, Mount K2 >
  < Adult human, No. of bones, 206 > **->** < Adult human, No. of bones, 300 >
  < Water, Boiling point, 100 °C > **->** < Water, Boiling point, 200 °C >

Conditions for Successful Edit

Condition 1: Reliability : It expects $f_{edit}(\theta')$ to select y_{alt} instead of y_{old} for a given input triplet t from D_x where D_x is the set of targeted queries

Example:

For t : < The US, President, Barack Obama > to < The US, President, Joe Biden >

Input query : The president of the US is ...
Expected output : ~~Barack Obama~~ Joe Biden

Conditions for Successful Edit

Condition 2: Generalization: It expects $f_{edit}(\theta')$ to select y_{alt} instead of y_{old} for the paraphrased versions of inputs from D_x, denoted by P_x

Example:

For t : < The US, President, Barack Obama > to < The US, President, Joe Biden >

Input query : The president of the US is
Expected output : ~~Barack Obama~~ Joe Biden

Input query : Who is the president of the US?
Expected output : ~~Barack Obama~~ Joe Biden

Conditions for Successful Edit

Condition 3: Localization: It expects $f_{edit}(\theta')$ to select y_{old} instead of y_{alt} for a given input triplet t from O_x where O_x is the set of non-targeted queries.

Example:

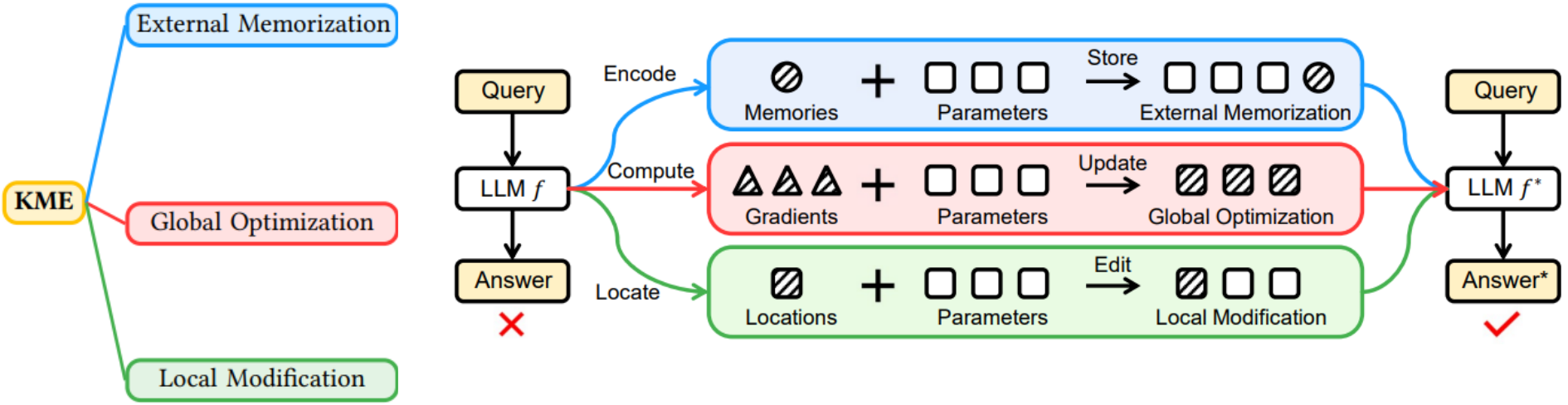
For t : < Russia, President, Vladimir Putin>

Input query : The president of Russia is
Expected output : Vladimir Putin

Input query : Who is the president of Russia?
Expected output : Vladimir Putin

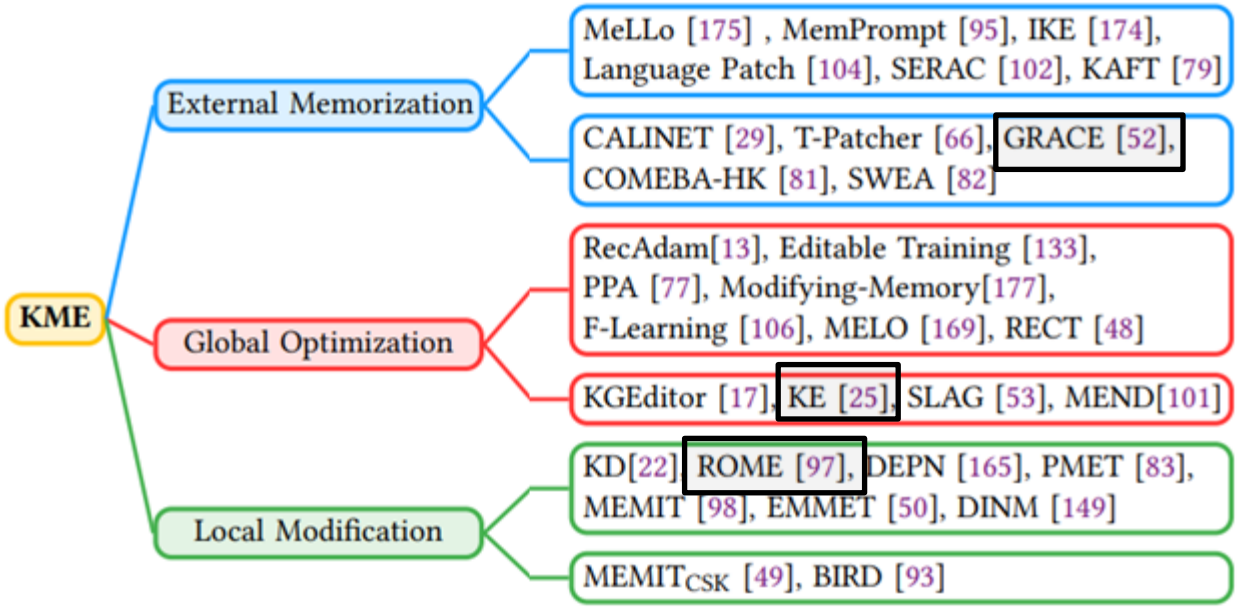
Taxonomy

Where? (e.g., external parameters or internal weights)
How? (e.g., via optimization or direct incorporation)



https://arxiv.org/pdf/2310.16218

Taxonomy



https://arxiv.org/pdf/2310.16218

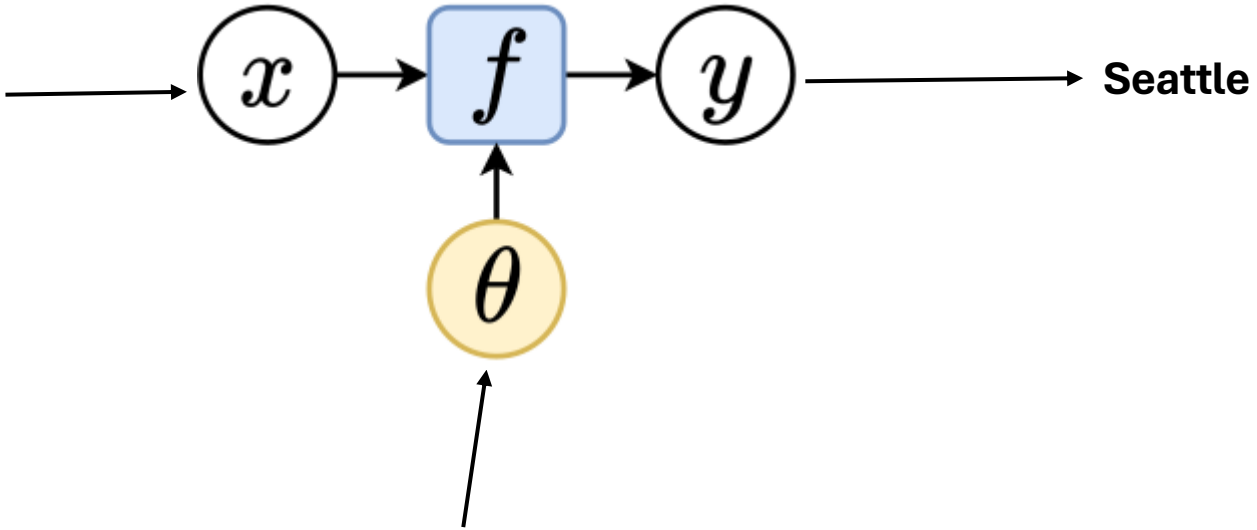
KnowledgeEditor

Editing Factual Knowledge in Language Models

Nicola De Cao [1,2], Wilker Aziz [1], Ivan Titov [1,2]
[1]University of Amsterdam, [2]University of Edinburgh
{ nicola.decao, w.aziz, titov } @uva.nl

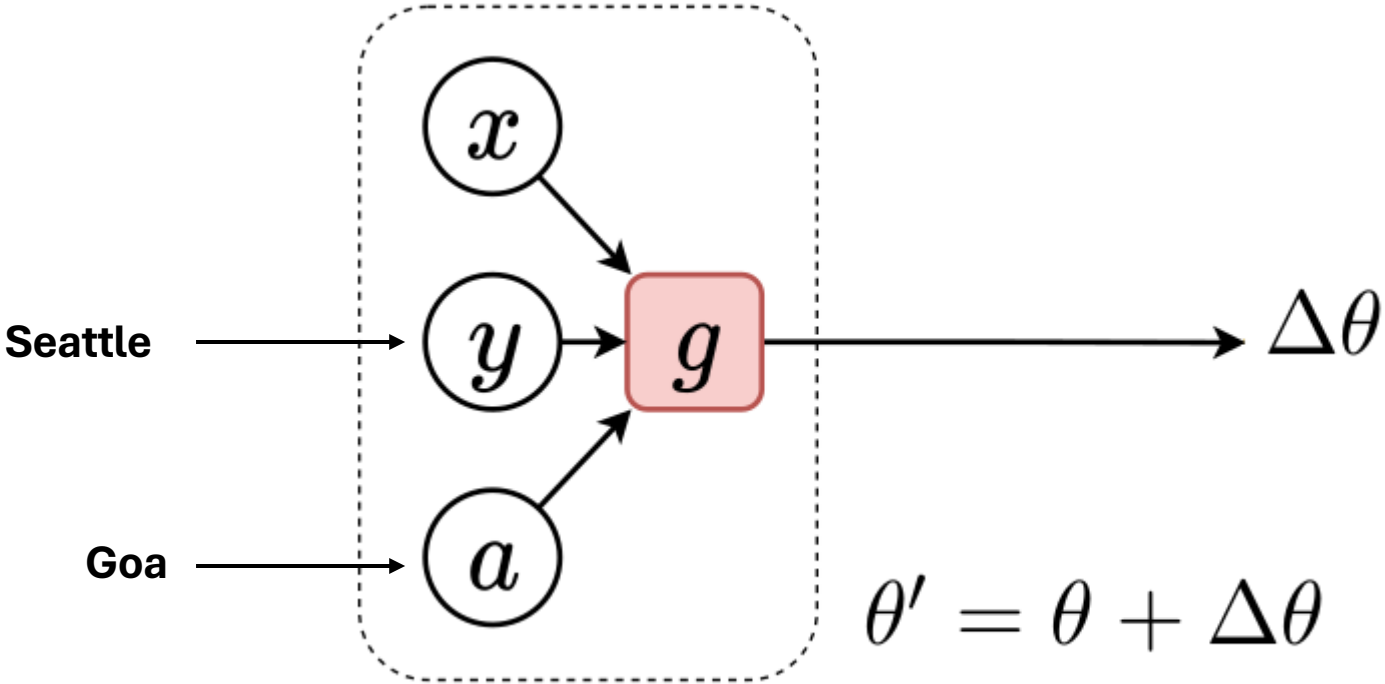
Method: KNOWLEDGE EDITOR

The Space Needle is located in the city of

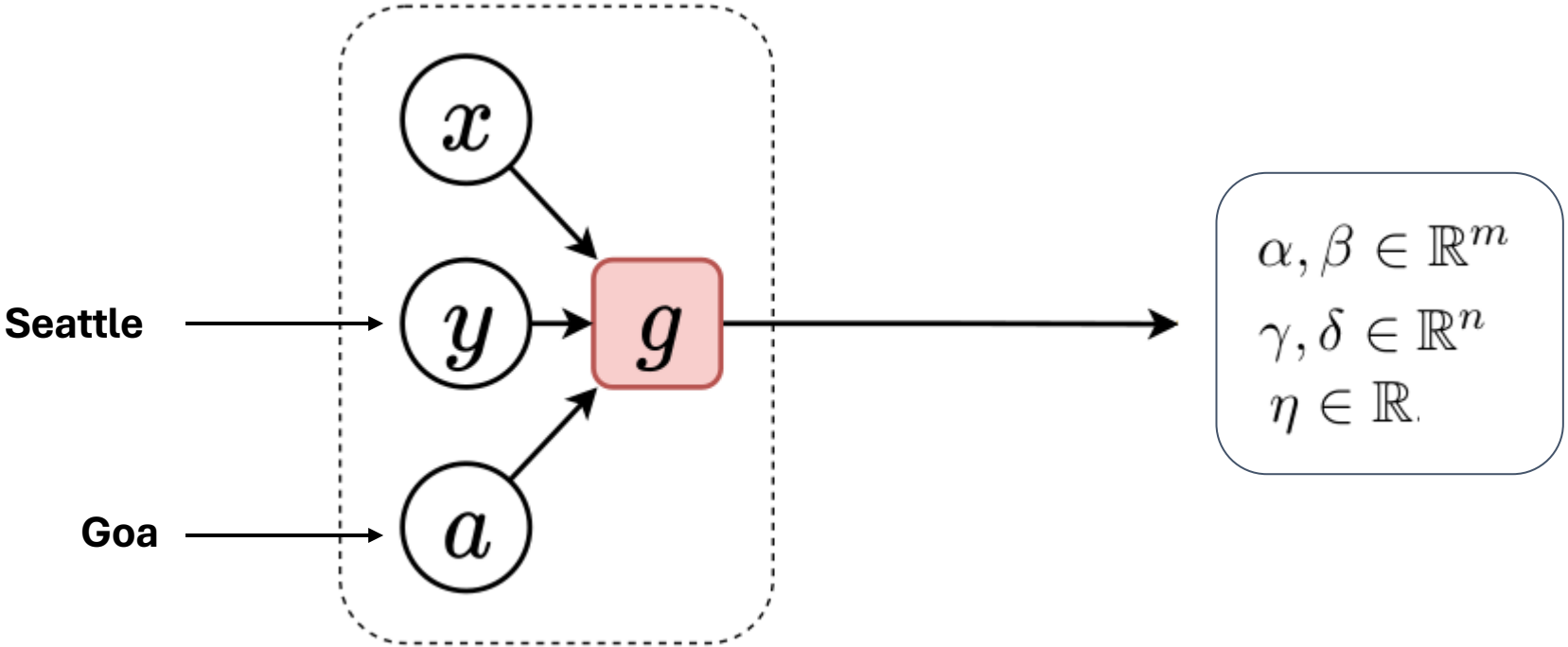


The parameters of a **pre-trained** model

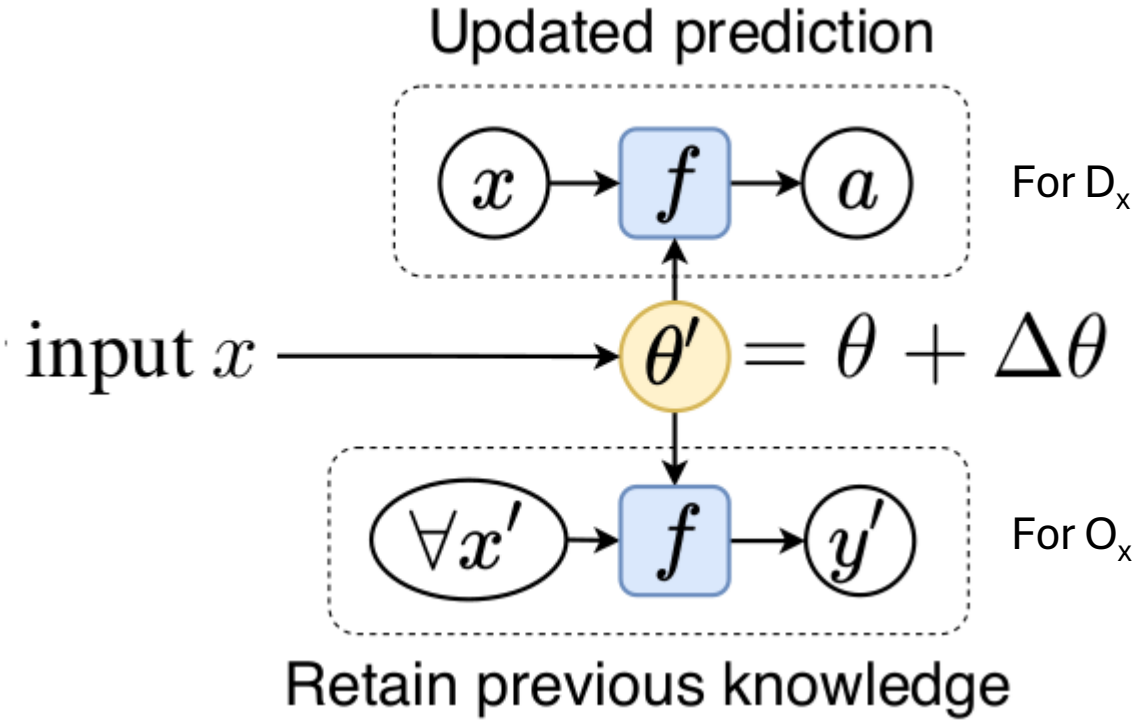
Method: KNOWLEDGE EDITOR



Method: KNOWLEDGE EDITOR



Method: KNOWLEDGE EDITOR



Training Hyper-network

$$\min_{\phi} \quad \sum_{\hat{x} \in \mathcal{P}^x} \mathcal{L}(\theta'; \hat{x}, a)$$

$$\text{s.t.} \quad \mathcal{C}(\theta, \theta', f; \mathcal{O}^x) < m$$

$$\mathcal{C}_{KL}(\theta, \theta', f; \mathcal{O}^x) = \sum_{x' \in \mathcal{O}^x} \sum_{c \in \mathcal{Y}} p_{Y|X}(c|x', \theta) \log \frac{p_{Y|X}(c|x', \theta)}{p_{Y|X}(c|x', \theta')}$$

$$\mathcal{C}_{L_p}(\theta, \theta', f; \mathcal{O}^x) = \left(\sum_i |\theta_i - \theta'_i|^p\right)^{1/p}$$

Training Hyper-network

$$\min_{\phi} \quad \sum_{\hat{x} \in \mathcal{P}^x} \mathcal{L}(\theta'; \hat{x}, a)$$

$$\text{s.t.} \quad \mathcal{C}(\theta, \theta', f; \mathcal{O}^x) < m$$

$$\mathcal{C}_{KL}(\theta, \theta', f; \mathcal{O}^x) = \sum_{x' \in \mathcal{O}^x} \sum_{c \in \mathcal{Y}} p_{Y|X}(c|x', \theta) \log \frac{p_{Y|X}(c|x', \theta)}{p_{Y|X}(c|x', \theta')}$$

$$\mathcal{C}_{L_p}(\theta, \theta', f; \mathcal{O}^x) = \left(\sum_i |\theta_i - \theta'_i|^p\right)^{1/p}$$

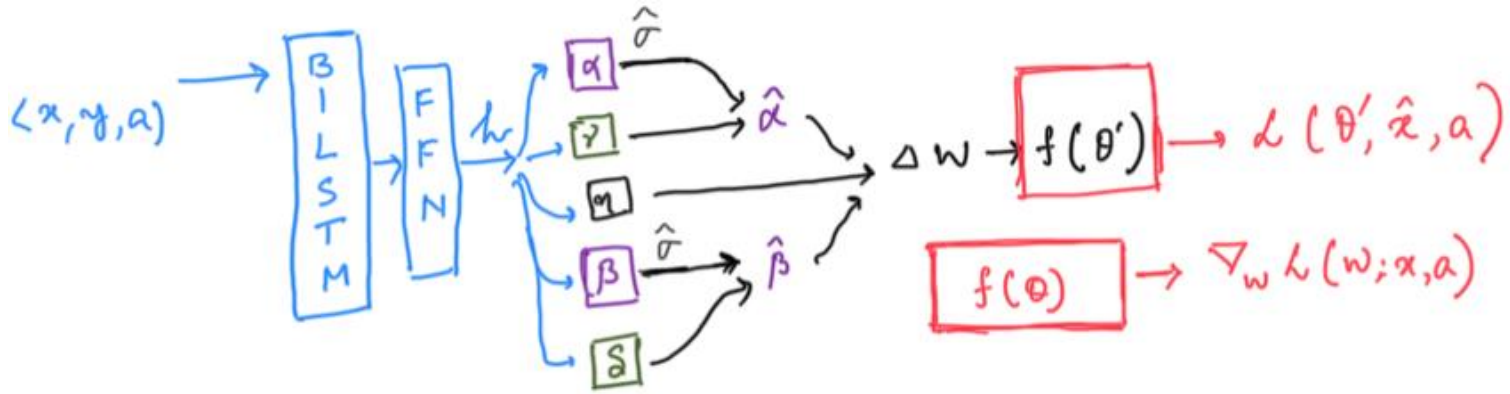
$$\Delta W = \sigma(\eta) \cdot \left(\hat{\alpha} \odot \nabla_W \mathcal{L}(W; x, a) + \hat{\beta}\right) ,$$

$$\text{with} \quad \hat{\alpha} = \hat{\sigma}(\alpha)\gamma^\top \quad \text{and} \quad \hat{\beta} = \hat{\sigma}(\beta)\delta^\top ,$$

$\hat{\sigma}$: Softmax

σ : Sigmoid

Training Hyper-network



$$\min_{\phi} \quad \sum_{\hat{x} \in \mathcal{P}^x} \mathcal{L}(\theta'; \hat{x}, a)$$

$$\text{s.t.} \quad \mathcal{C}(\theta, \theta', f; \mathcal{O}^x) < m$$

$$\mathcal{C}_{KL}(\theta, \theta', f; \mathcal{O}^x) = \sum_{x' \in \mathcal{O}^x} \sum_{c \in \mathcal{Y}} p_{Y|X}(c|x', \theta) \log \frac{p_{Y|X}(c|x', \theta)}{p_{Y|X}(c|x', \theta')}$$

$$\mathcal{C}_{L_p}(\theta, \theta', f; \mathcal{O}^x) = \left(\sum_i |\theta_i - \theta'_i|^p\right)^{1/p}$$

$$\Delta W = \sigma(\eta) \cdot \left(\hat{\alpha} \odot \nabla_W \mathcal{L}(W; x, a) + \hat{\beta}\right),$$

$$\text{with} \quad \hat{\alpha} = \hat{\sigma}(\alpha)\gamma^\top \quad \text{and} \quad \hat{\beta} = \hat{\sigma}(\beta)\delta^\top,$$

$\hat{\sigma}$: Softmax

σ : Sigmoid

Evaluation: KNOWLEDGE EDITOR

BERT-base on FEVER dataset

Method	Fact-Checking		
	Success rate ↑	Retain acc ↑	Equiv. acc ↑
Fine-tune (1st layer)	100.0	99.44	42.24
Fine-tune (all layers)	100.0	86.95	95.58
Zhu et al. (1st layer)	100.0	99.44	40.30
Zhu et al. (all layers)	100.0	94.07	83.30
Ours $\mathcal{C}_{L_2}$	99.10	45.10	99.01
KNOWLEDGEEDITOR	98.80	98.14	82.69
+ loop†	100.0	97.78	81.57
+ $\mathcal{P}^x$ ‡	98.50	98.55	95.25
+ $\mathcal{P}^x$ + loop‡	100.0	98.46	94.65

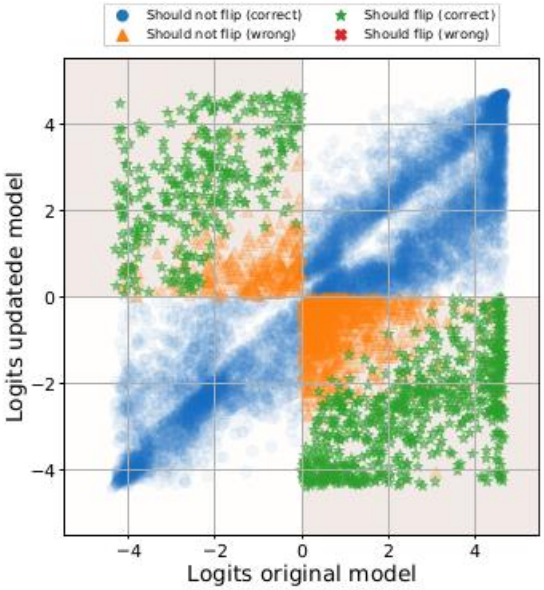
success rate: how much g successfully updates the knowledge in θ', measured as accuracy of revised predictions for inputs in $\mathcal{D}$;

retain accuracy: how well θ' retains the original predictions of f, measured as accuracy wrt input-output pairs in sets $\mathcal{O}^x$;

equivalence accuracy: how consistent the predictions of the revised model θ' are for semantically equivalent inputs, measured as accuracy of the revised predictions for all $\mathcal{P}^x$;

Evaluation: KNOWLEDGE EDITOR

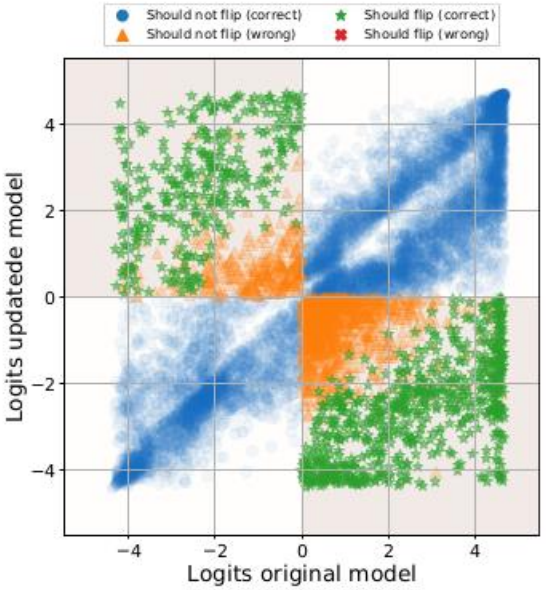
(a) Fine-tune (all layers)



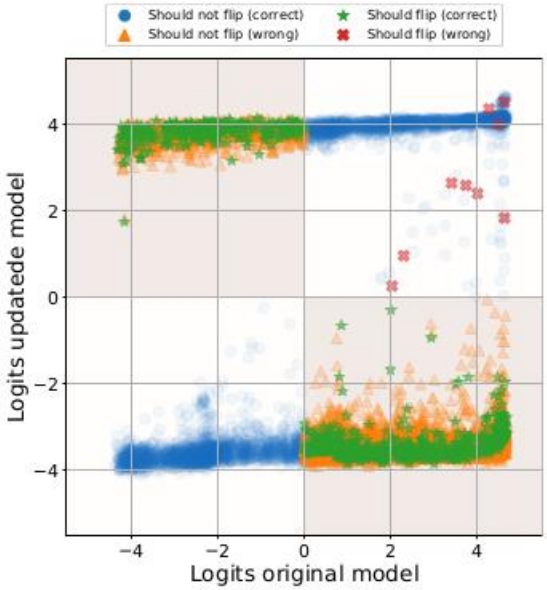
Distribution of logits in the original model and the updated model on FEVER dataset. (a) Fine-tune all layers causes ample amount of errors. (b) C_{L2} is able to alter the predictions but fails at retaining the updates. (c) KNOWLEDGE - EDITOR with C_{KL}, P_x has minimal errors and is able to achieve correct output with a small perturbation.

Evaluation: KNOWLEDGE EDITOR

(a) Fine-tune (all layers)



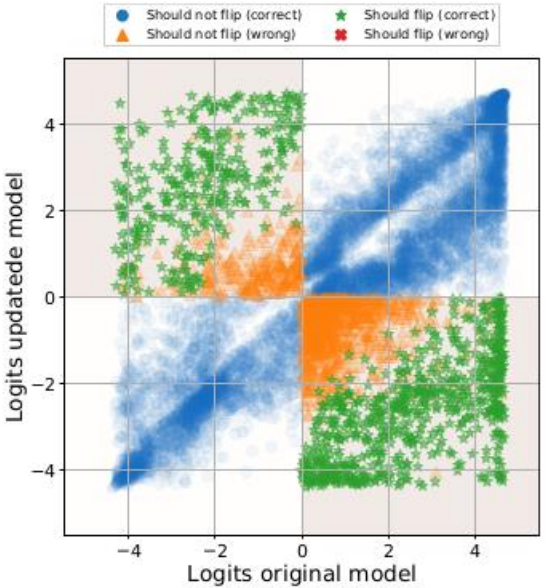
(b) C_{L2}



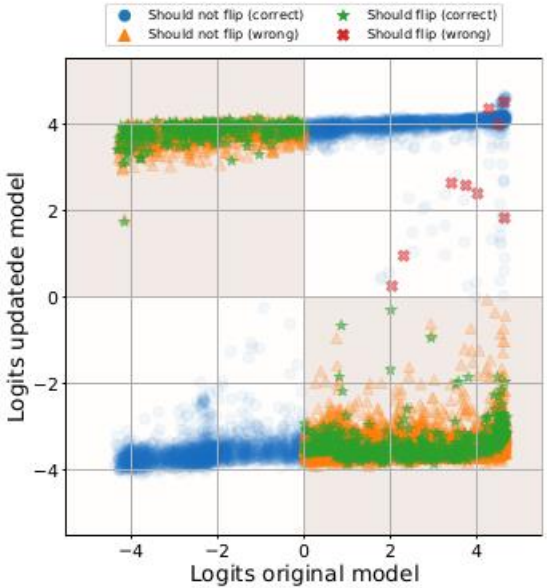
Distribution of logits in the original model and the updated model on FEVER dataset. (a) Fine-tune all layers causes ample amount of errors. (b) C_{L2} is able to alter the predictions but fails at retaining the updates. (c) KNOWLEDGE - EDITOR with C_{KL}, P_x has minimal errors and is able to achieve correct output with a small perturbation.

Evaluation: KNOWLEDGE EDITOR

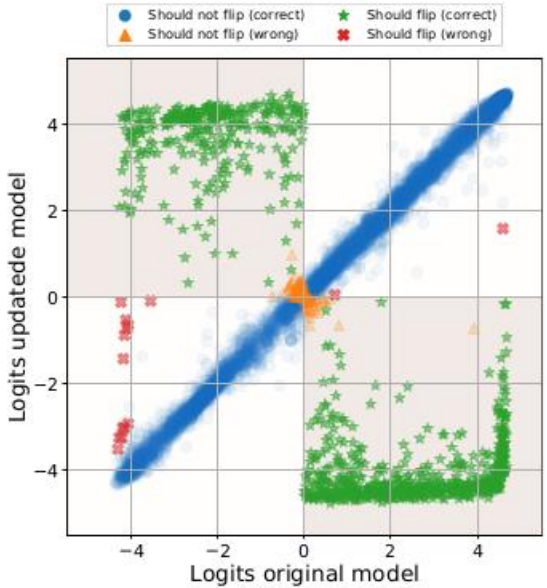
(a) Fine-tune (all layers)



(b) C_{L2}



(c) KNOWLEDGE-EDITOR with C_{KL} + P_x



Distribution of logits in the original model and the updated model on FEVER dataset. (a) Fine-tune all layers causes ample amount of errors. (b) C_{L2} is able to alter the predictions but fails at retaining the updates. (c) KNOWLEDGE - EDITOR with C_{KL}, P_x has minimal errors and is able to achieve correct output with a small perturbation.

GRACE

Aging with GRACE: Lifelong Model Editing with Discrete Key-Value Adaptors

Thomas Hartvigsen
University of Virginia, MIT
hartvigsen@virginia.edu

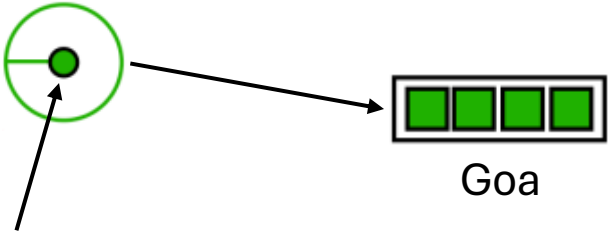
Swami Sankaranarayanan
Sony AI
swami.sankaranarayanan@sony.com

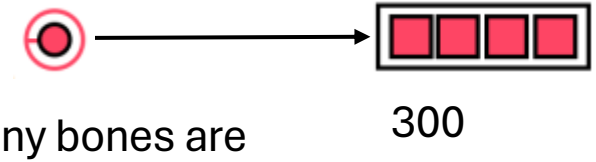
Hamid Palangi
Microsoft Research
hpalangi@microsoft.com

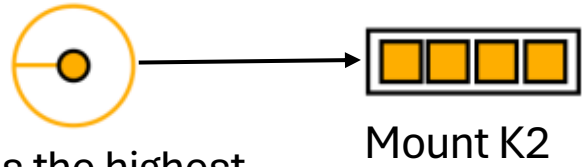
Yoon Kim
MIT
yoonkim@mit.edu

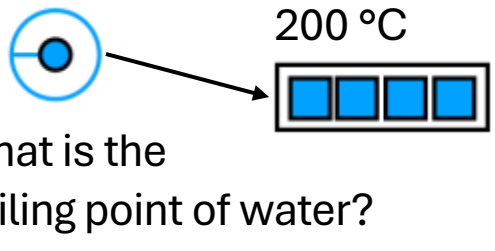
Marzyeh Ghassemi
MIT
mghassem@mit.edu

Method: GRACE









GRACE Adaptor

- Uses a cache-like storage for a layer, called **GRACE Adaptor**

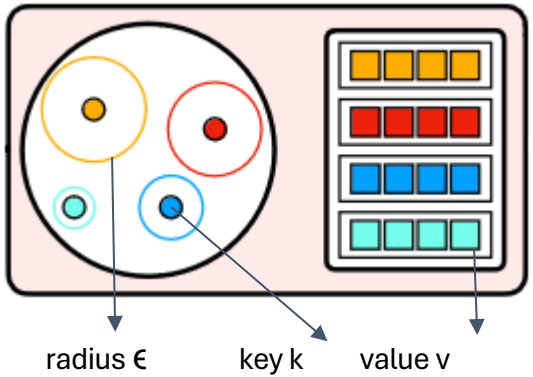
- An Adaptor contains **two** components:

  (1) a codebook **C**

  (2) a deferral mechanism

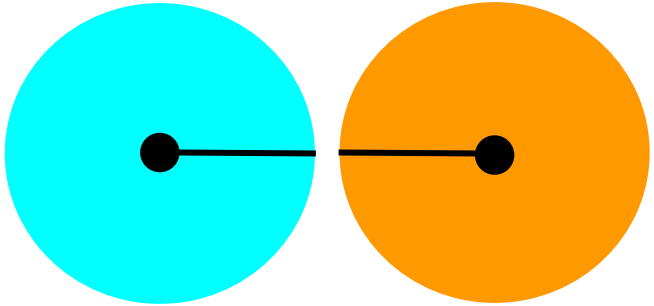
GRACE: Codebook

- The codebook **C** has three components
  - A set of keys (K)
  - A set of values (V)
  - Deferral radii (ξ)



GRACE: Deferral Mechanism

- In case of **no match** or **empty** codebook **C**
- A new entry will be created.



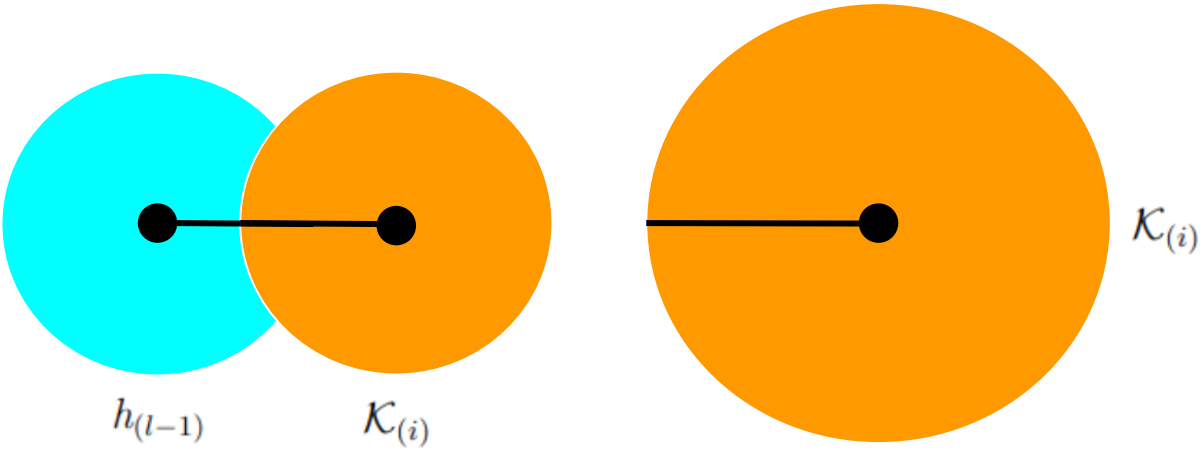
$$h_{(l)} = \begin{cases} \text{GRACE}(h_{(l-1)}), & \text{if } min_i(d(h_{(l-1)}, \mathcal{K}_{(i)})\ ) < \xi_{(i)} \\ f_l(h_{(l-1)}), & \text{otherwise} \end{cases}$$

Algorithm 1: Update Codebook at layer l.

Input: $\mathcal{C} = \{(\mathbb{K}_i, \mathbb{V}_i, \epsilon_i)\}_{i=0}^{C-1}$, codebook
Input: $f(\cdot)$, model
Input: y_t, desired label
Input: x_t, edit input for which $f(x_t) \neq y_t$
Input: ϵ_{init}, initial ϵ
Input: $\text{d}(\cdot)$, distance function
Output: $\mathcal{C}$, updated codebook
$C = \|\mathcal{C}\|$
$\hat{y}, h^{l-1} = f^L(x_t), f^{l-1}(x_t)$
$\text{d}_{\text{min}}, i = \min_i(d(h^{l-1}, \mathbb{K}_i))$
If $\text{d}_{\text{min}} > \epsilon_i + \epsilon_{\text{init}}$ or $C = 0$:
  # h^{l-1} far from existing entries or empty $\mathcal{C}$
  $v_{\text{new}} = \text{finetune on } P_f(y|v_{\text{init}})$
  $\mathcal{C}_C = (h^{l-1}, v_{\text{new}}, \epsilon_{\text{init}})$ # *Add entry*

GRACE: Deferral Mecganism

- In case of **collision**, expand if the o* is **same**.



$$h_{(l)} = \begin{cases} \text{GRACE}(h_{(l-1)}), & \text{if } min_i(d(h_{(l-1)}, \mathcal{K}_{(i)})) < \xi_{(i)} \\ f_l(h_{(l-1)}), & \text{otherwise} \end{cases}$$

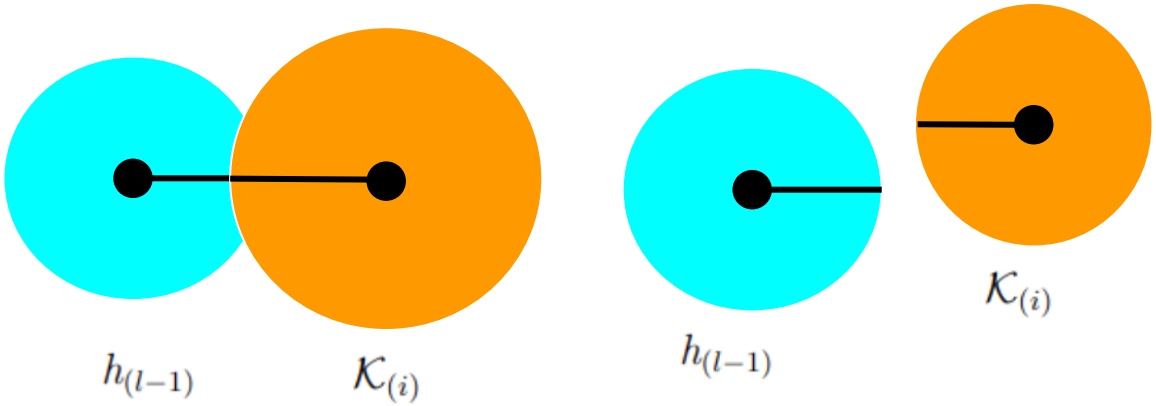
Algorithm 1: Update Codebook at layer l.

Input: $\mathcal{C} = \{(\mathbb{K}_i, \mathbb{V}_i, \epsilon_i)\}_{i=0}^{C-1}$, codebook
Input: $f(\cdot)$, model
Input: y_t, desired label
Input: x_t, edit input for which $f(x_t) \neq y_t$
Input: ϵ_{init}, initial ϵ
Input: $\text{d}(\cdot)$, distance function
Output: $\mathcal{C}$, updated codebook

$C = \|\mathcal{C}\|$
$\hat{y}, h^{l-1} = f^L(x_t), f^{l-1}(x_t)$
$\text{d}_{\text{min}}, i = \min_i(d(h^{l-1}, \mathbb{K}_i))$
If $\text{d}_{\text{min}} > \epsilon_i + \epsilon_{\text{init}}$ or $C = 0$:
  # h^{l-1} far from existing entries or empty $\mathcal{C}$
  $v_{\text{new}} = \text{finetune on } P_f(y|v_{\text{init}})$
  $\mathcal{C}_C = (h^{l-1}, v_{\text{new}}, \epsilon_{\text{init}})$ # *Add entry*
Else:
  # h^{l-1} near existing entries
  If $f^L(k_i) = y$:
    # *Same label → Expand*
    $\mathcal{C}_i := (k_i, v_i, \epsilon_i + \epsilon_{\text{init}})$

GRACE: Deferral Mecganism

- In case of **collision**, reduce if the o* is **different**.



$$h_{(l)} = \begin{cases} \text{GRACE}(h_{(l-1)}), & \text{if } min_i(d(h_{(l-1)}, \mathcal{K}_{(i)})\ ) < \xi_{(i)} \\ f_l(h_{(l-1)}), & \text{otherwise} \end{cases}$$

Algorithm 1: Update Codebook at layer l.

Input: $\mathcal{C} = \{(\mathbb{K}_i, \mathbb{V}_i, \epsilon_i)\}_{i=0}^{C-1}$, codebook
Input: $f(\cdot)$, model
Input: y_t, desired label
Input: x_t, edit input for which $f(x_t) \neq y_t$
Input: ϵ_{init}, initial ϵ
Input: $\text{d}(\cdot)$, distance function
Output: $\mathcal{C}$, updated codebook

$C = \|\mathcal{C}\|$
$\hat{y}, h^{l-1} = f^L(x_t), f^{l-1}(x_t)$
$\text{d}_{\text{min}}, i = \min_i(d(h^{l-1}, \mathbb{K}_i))$
If $\text{d}_{\text{min}} > \epsilon_i + \epsilon_{\text{init}}$ or $C = 0$:
  # h^{l-1} far from existing entries or empty $\mathcal{C}$
  $v_{\text{new}} = \text{finetune on } P_f(y|v_{\text{init}})$
  $\mathcal{C}_C = (h^{l-1}, v_{\text{new}}, \epsilon_{\text{init}})$ # *Add entry*
Else:
  # h^{l-1} near existing entries
  If $f^L(k_i) = y$:
    # *Same label → Expand*
    $\mathcal{C}_i := (k_i, v_i, \epsilon_i + \epsilon_{\text{init}})$
  Else:
    # *Different label → Split*
    $\mathcal{C}_i = (k_i, v_i, \text{d}_{\text{min}}/2)$ # *Update entry i*
    $v_{\text{new}} = \text{finetune on } P_f(y|v_{\text{init}})$
    $\mathcal{C}_C = (h^{l-1}, v_{\text{new}}, \text{d}_{\text{min}}/2)$ # *Add entry*
return: $\mathcal{C}$

GRACE: Evaluation

Method	zsRE (T5; F1 ↑)				SCOTUS (BERT; Acc ↑)			
	TRR	ERR	*Avg.*	*#E*	TRR	ERR	*Avg.*	*#E*
FT [25]	.56	.82	*.69*	1000	.52	.52	*.52*	415
FT+EWC [19]	.51	.82	*.66*	1000	.67	.50	*.58*	408
FT+Retrain [36]	.27	.99	*.63*	1000	.67	**.83**	*.75*	403
MEND [30]	.25	.27	*.26*	1000	.19	.27	*.23*	672
Defer [31]	**.72**	.31	*.52*	1000	.33	.41	*.37*	506
ROME [28]	—	—	—	—	—	—	—	—
Memory	.25	.27	*.26*	1000	.21	.20	*.21*	780
GRACE	.69	**.96**	***.82***	1000	**.81**	.82	***.82***	381

#E: the number of edits
#TRR: it evaluates the knowledge retention rate of the edited model.
#ERR: it evaluates the retention rate of old-edits.

- T5-60M, task: context-free QA, zSRE dataset.
- BERT-110M, task: label shifts in legal documents in US, SCOTUS dataset.

ROME

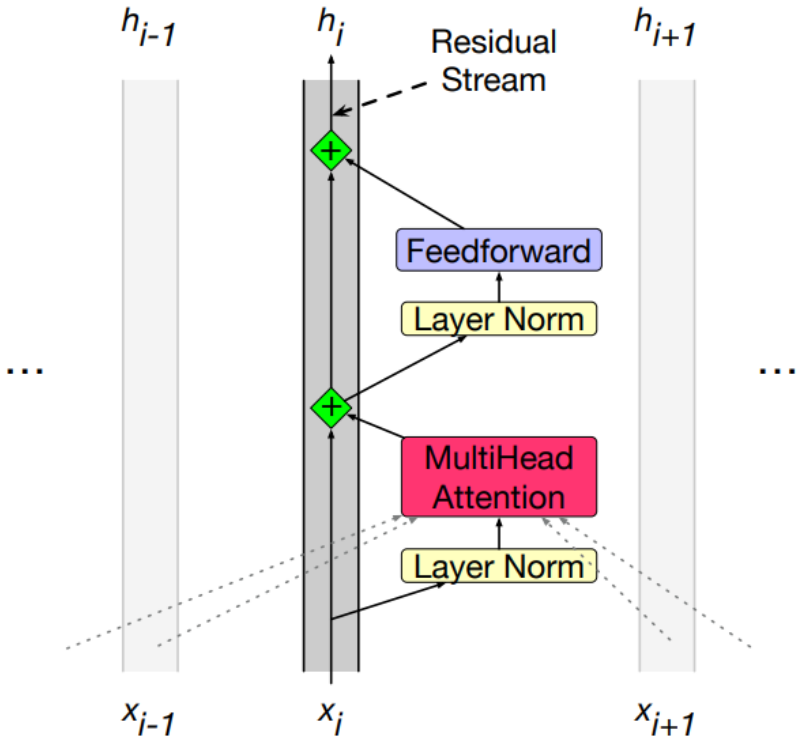
Locating and Editing Factual Associations in GPT

Kevin Meng* MIT CSAIL
David Bau* Northeastern University
Alex Andonian MIT CSAIL
Yonatan Belinkov† Technion – IIT

Method: ROME

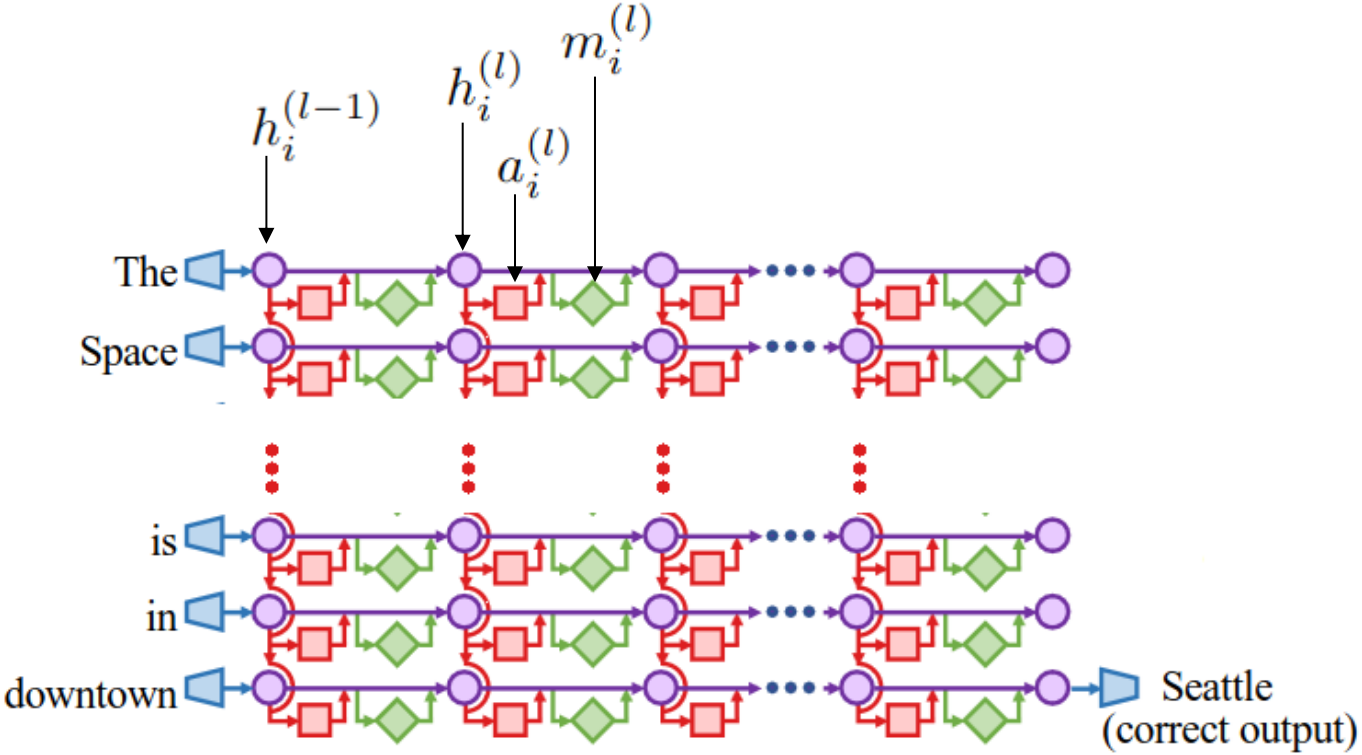
- An in-depth analysis on the storage and recall of factual associations in autoregressive transformer language models.
- It provides evidence that factual associations correspond to localized, directly-editable computations.
- It proposes a mechanism to update specific factual associations in feed-forward weights using **ROME**.
- First to find that the mid-layer feedforward modules play a crucial role in storing factual associations and it can be altered with a direct manipulation.

Residual Stream Perspective of Transformer



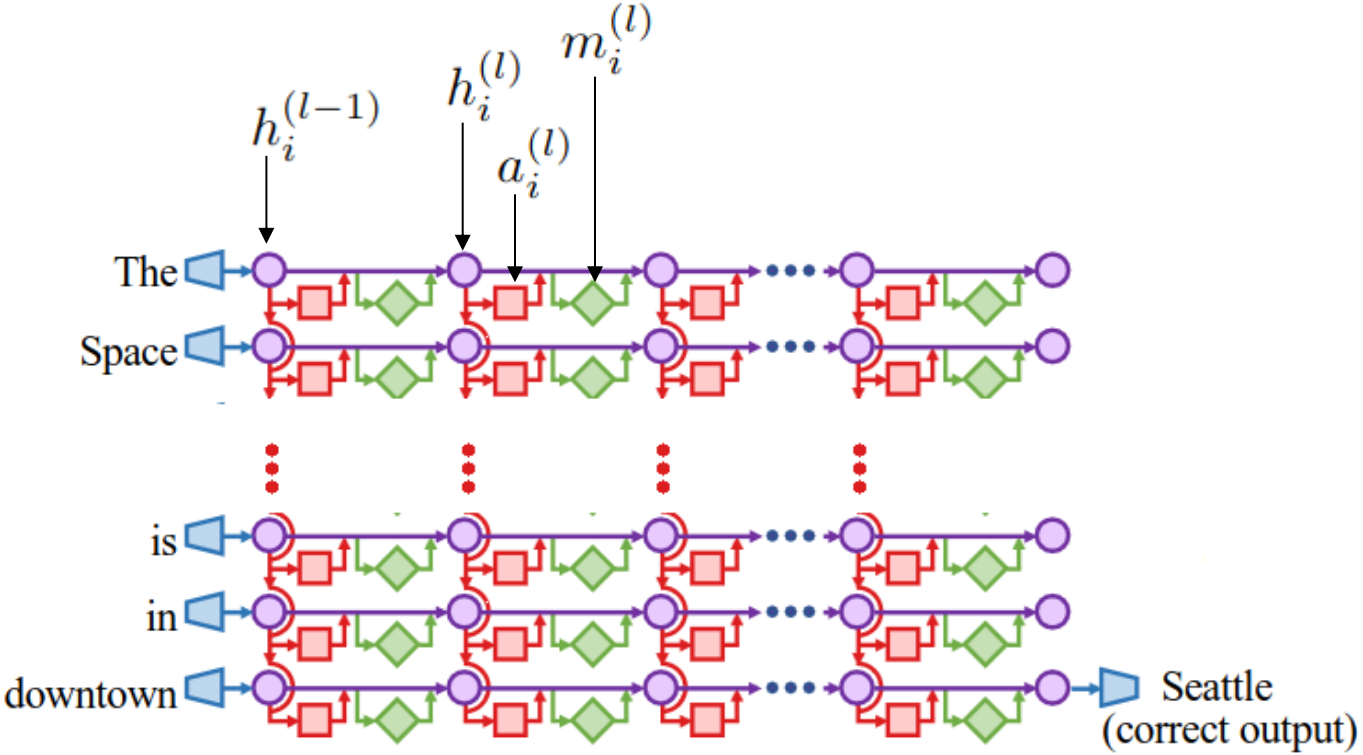
https://web.stanford.edu/~jurafsky/slp3/9.pdf

Method: ROME


$h_i^{(l)}$ state
attention $a_i^{(l)}$
MLP
$h_i^{(l-1)}$
$h_i^{(l)}$
$a_i^{(l)}$
$m_i^{(l)}$
The
Space
is
in
downtown
Seattle
(correct output)

Method: ROME

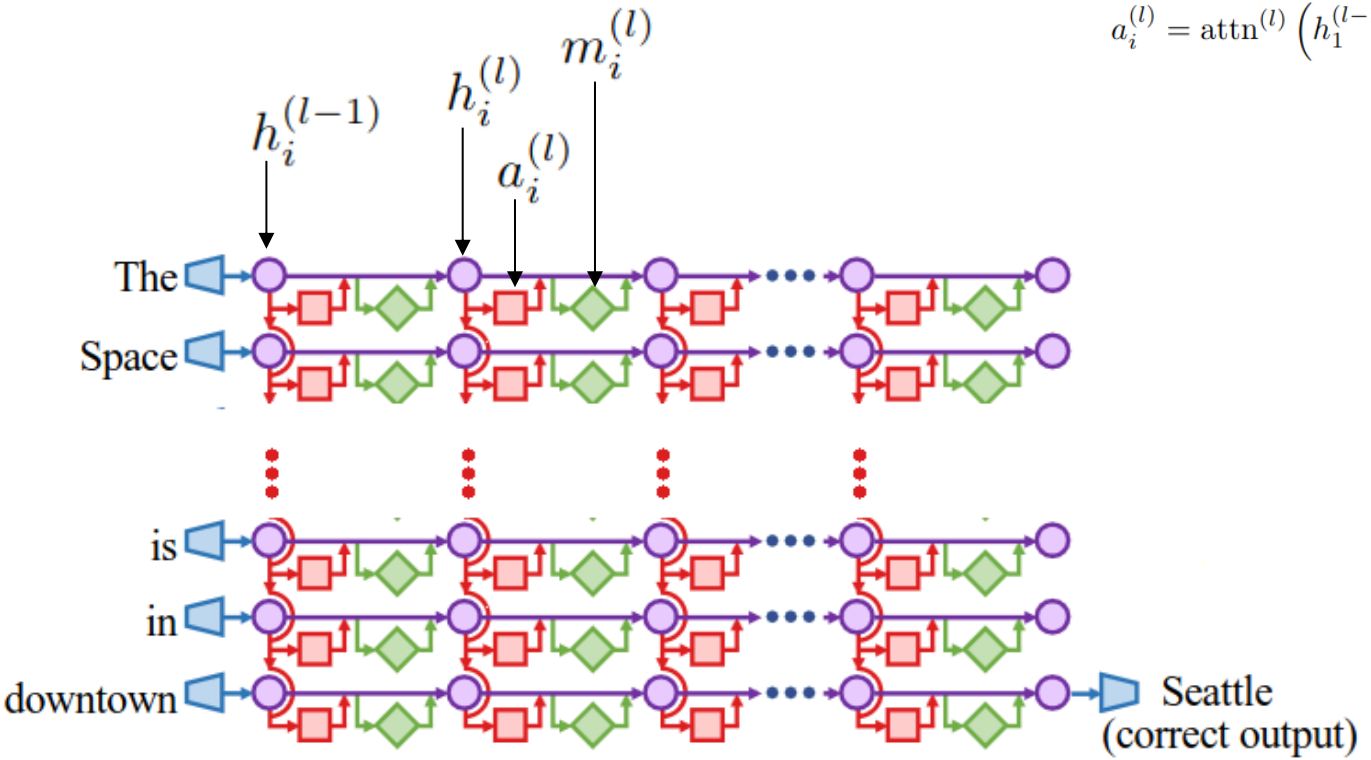
$$h_i^{(l)} = h_i^{(l-1)} + a_i^{(l)} + m_i^{(l)}$$



Method: ROME

$$h_i^{(l)} = h_i^{(l-1)} + a_i^{(l)} + m_i^{(l)}$$

$$a_i^{(l)} = \text{attn}^{(l)}\left(h_1^{(l-1)}, h_2^{(l-1)}, \ldots, h_i^{(l-1)}\right)$$

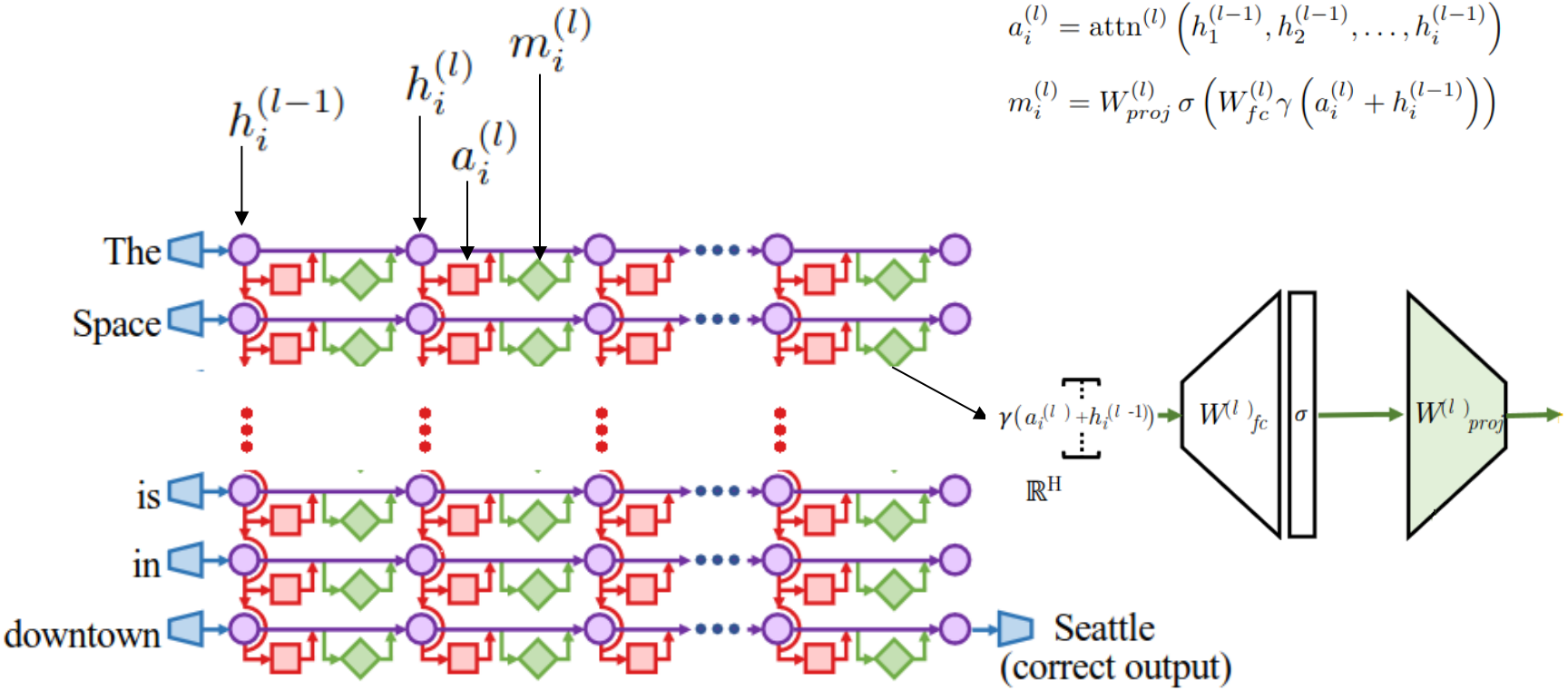


Method: ROME

$$h_i^{(l)} = h_i^{(l-1)} + a_i^{(l)} + m_i^{(l)}$$

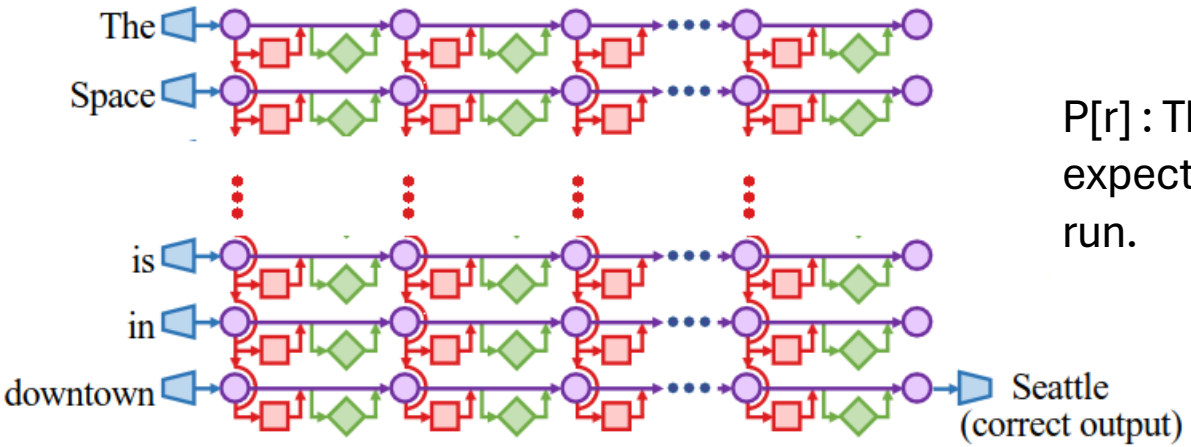
$$a_i^{(l)} = \text{attn}^{(l)}\left(h_1^{(l-1)}, h_2^{(l-1)}, \ldots, h_i^{(l-1)}\right)$$

$$m_i^{(l)} = W_{proj}^{(l)}\,\sigma\left(W_{fc}^{(l)}\gamma\left(a_i^{(l)} + h_i^{(l-1)}\right)\right)$$



Method: ROME

A **clean** run

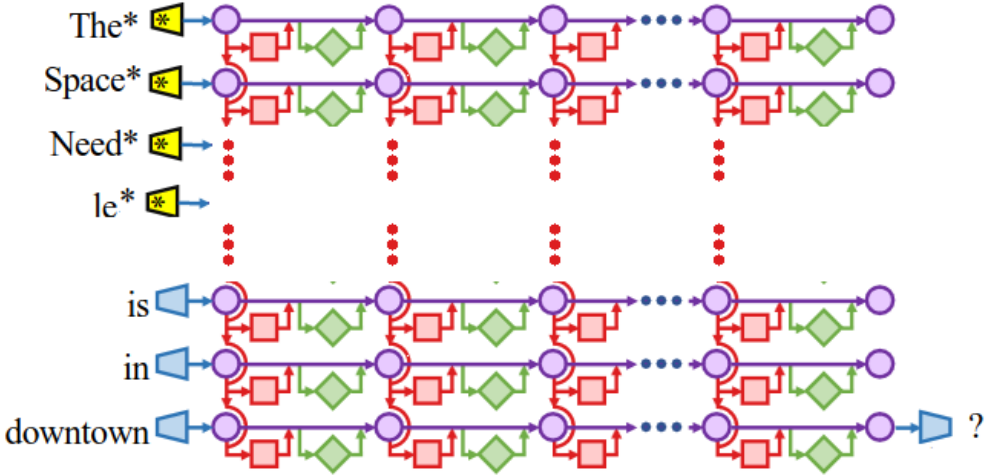


P[r] : The probability of emitting the expected response r under the **clean** run.

A **corrupted** run

Method: ROME

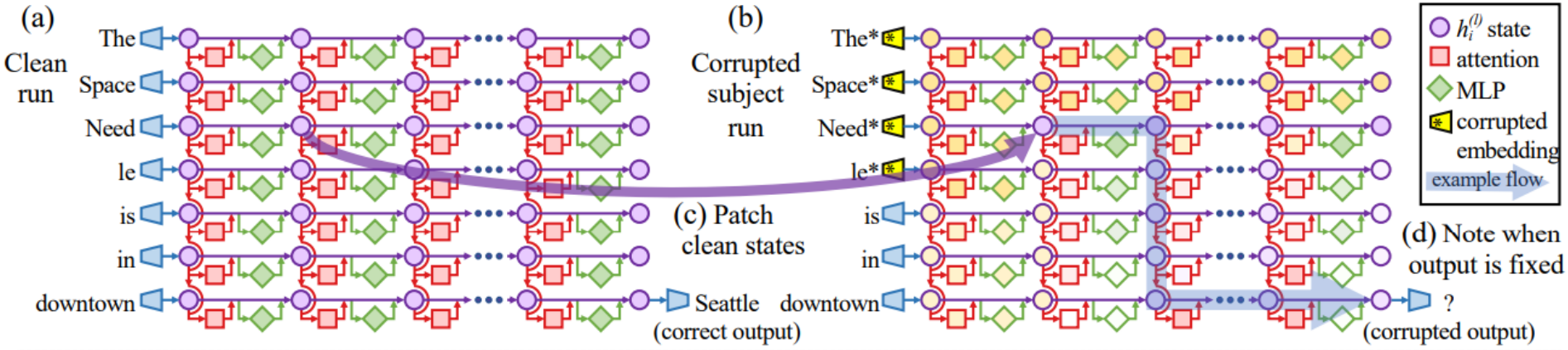
$$h_i'^{(0)} = h_i^{(0)} + \epsilon$$



$P_*[r]$: The probability of emitting the expected response r under the **corrupted** run.

Method: ROME

A **corrupted-with-restoration** run



Method: ROME

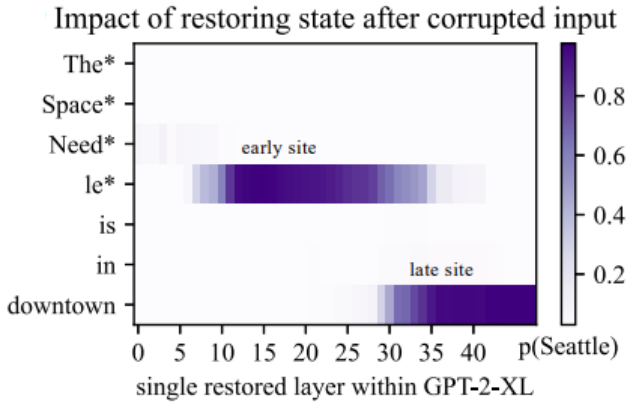
- The relative importance of a state s is approximated through the following process
  - **Total Effect** (TE): difference between the probabilities of generating the true response under clean run and corrupted run

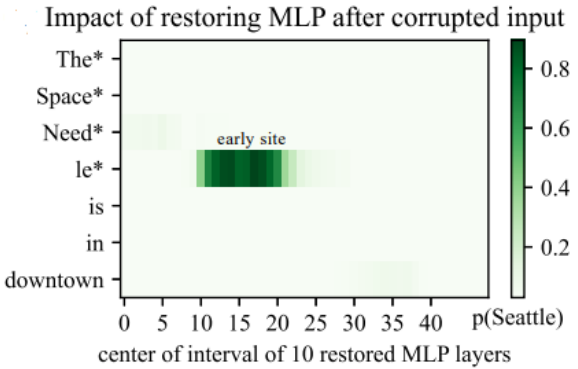
$$TE = P[r] - P_*[r]$$

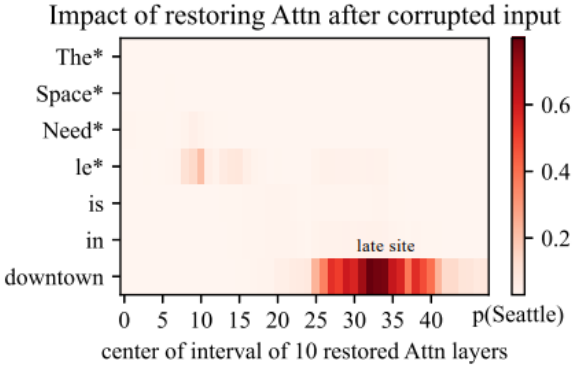
  - **Indirect Effect** (IE):

$$IE = P_*^{(k)}{}_i[r] - P_*[r]$$

Effect of Causal Tracing



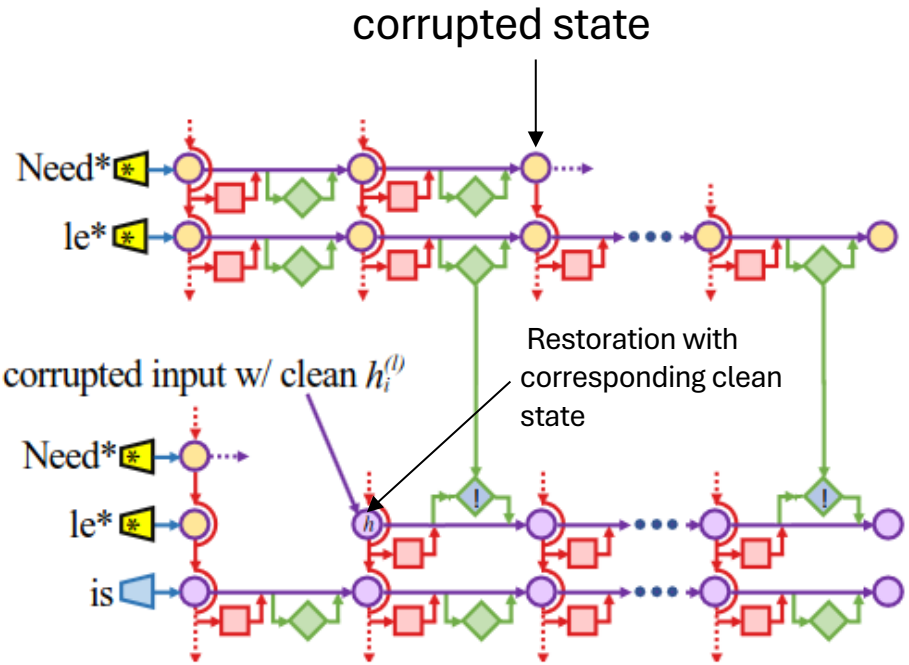




- **MLP** contributions dominate the **early** site with their contributions peaking at an AIE of 6.6%.
- **Attention** is important at the **late** site , with an AIE of 1.6% at the last subject token..

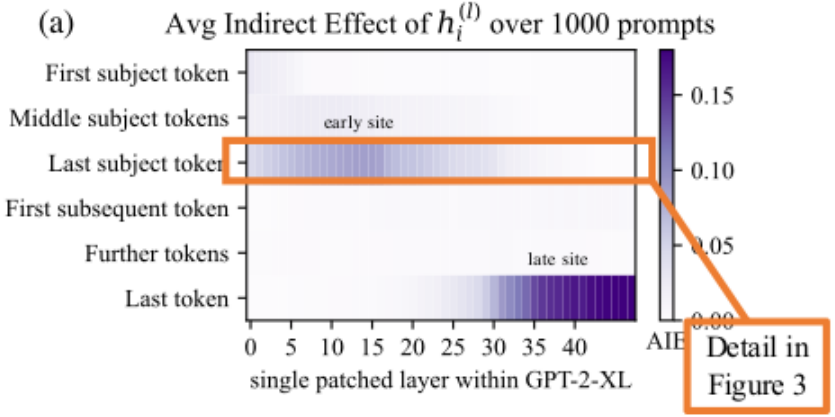
Method: ROME

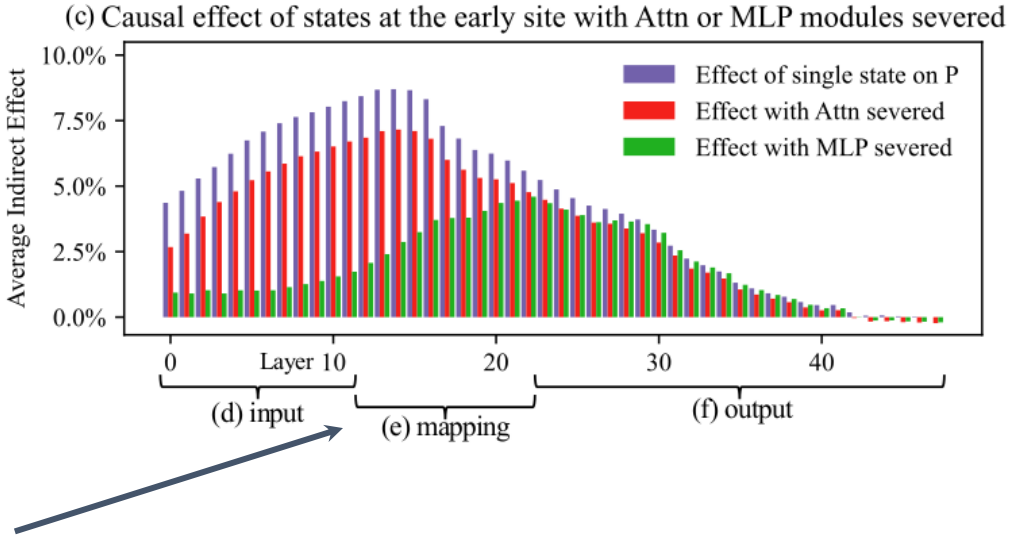
V.2: A **corrupted-with-restoration** run



$P_*^{(k)}{}_i[r]$: probability of emitting the true response r under the **corrupted-with-restoration** run with respect to the hidden state at the k-th layer and i-th token.

Method: ROME

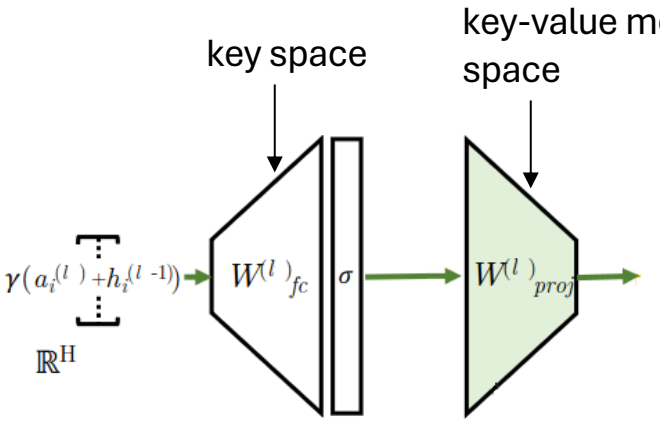




The localized mid-layer MLP key–value mapping recalls facts about the subject.

Method: ROME

- Geva et al. (EMNLP'21) observed that MLP layers c
  act as two-layer key–value memories.



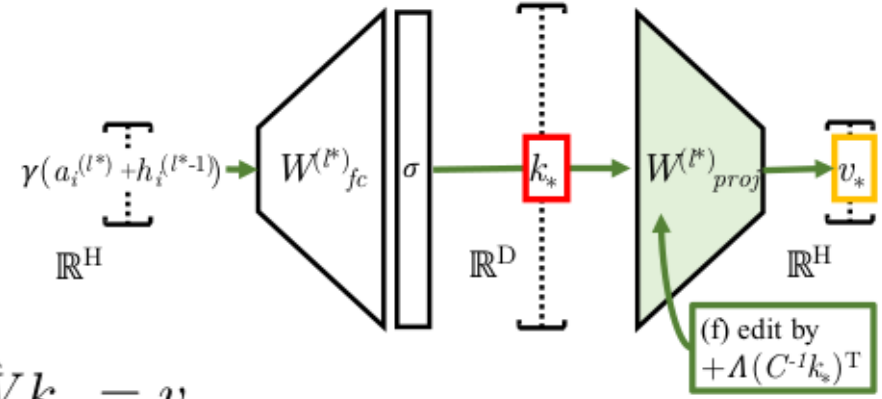
- Bau et al. (ECCV'20) observed that a new key–value pair (k^* , v^*) can be inserted optimally into the memory by solving a constrained least-squares problem.

Method: ROME

- The method ROME has **three** components
    - Key Selection
    - Value Encoding
    - Update and Save

Method: ROME

Objective



- A constrained least-squares problem

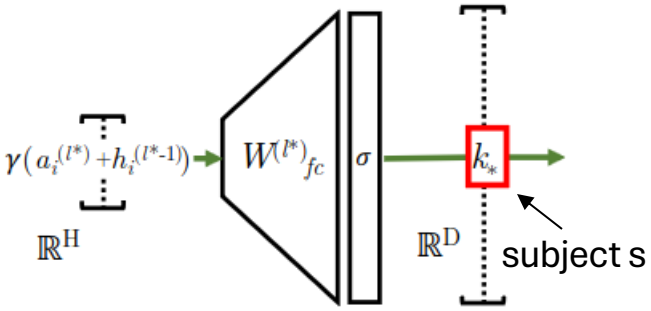
$$\text{minimize } \|\hat{W}K - V\| \text{ such that } \hat{W}k_* = v_*$$

$$\text{by setting } \hat{W} = W + \Lambda(C^{-1}k_*)^T$$

- W: MLP weights
- $C = KK^T$ where, K is a set of keys
- V is the set of corresponding updated values

- Λ is defined $(v_* - Wk_*)/(C^{-1}k_*)^T k_*$

Method: ROME



Key Selection

- The **last subject token** of the input is critical for a factual recall.

- The optimal key k* is computed over activations at layer L as

$$k_* = \frac{1}{N}\sum_{j=1}^{N} k(x_j + s), \text{where } k(x) = \sigma\left(W_{fc}^{(l^*)}\ \gamma(a_{[x],i}^{(l^*)} + h_{[x],i}^{(l^*-1)})\right)$$

  - k_* is set to an **average** value over a small set of texts containing the subject s.

Method: ROME

Value Encoding

- The value v_* is defined as $v_* = \text{argmin}_z \mathcal{L}(z)$; is $\mathcal{L}(z)$

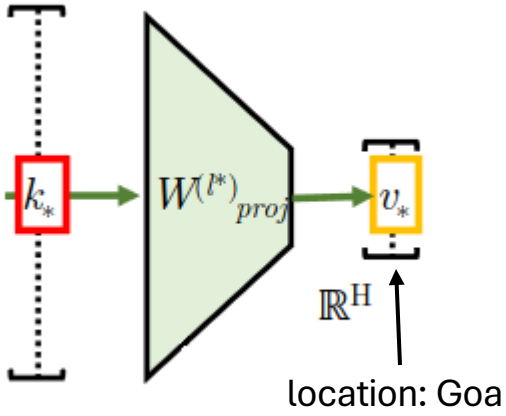
$$\frac{1}{N}\sum_{j=1}^{N} \underbrace{-\log \mathbb{P}_{G(m_i^{(l^*)}:=z)}\left[o^* \mid x_j + p\right]}_{\text{(a) Maximizing } o^* \text{ probability}} + \underbrace{D_{\mathrm{KL}}\left(\mathbb{P}_{G(m_{i'}^{(l^*)}:=z)}\left[x \mid p'\right] \middle\| \mathbb{P}_G\left[x \mid p'\right]\right)}_{\text{(b) Controlling essence drift}}$$

Loss 1

Loss 2

Method: ROME

Value Encoding



- Loss 1 is defined as,

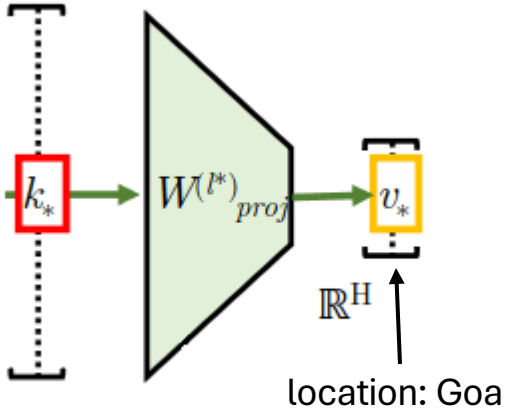
$$\frac{1}{N}\sum_{j=1}^{N}\underbrace{-\log \mathbb{P}_{G(m_i^{(l^*)}:=z)}\left[o^* \mid x_j + p\right]}_{\text{(a) Maximizing } o^* \text{ probability}}$$

- It finds a vector z that substituted as the output of the MLP for the token i which is the final part of the subject tokens.

- It will cause the network to predict the target object o^* in response to the factual prompt x.

Method: ROME

Value Encoding



- Loss 2 is defined as,

$$\underbrace{D_{\mathrm{KL}}\left(\mathbb{P}_{G(m_{i'}^{(l^*)}:=z)}\left[x \mid p'\right] \middle\| \mathbb{P}_{G}\left[x \mid p'\right]\right)}_{\text{(b) Controlling essence drift}}$$

- It minimizes the KL divergence of predictions for the prompt p' involving the same to the unchanged model.

- It preserve the model's understanding of the subject's essence.

ROME: Evaluation

Score	Efficacy		Generalization		Specificity		Fluency	Consistency
S ↑	ES ↑	EM ↑	PS ↑	PM ↑	NS ↑	NM ↑	GE ↑	RS ↑

- Score (E/P/N-S): It is the portion of cases for which P[o*] > P[o]
- Magnitude (E/P/N-M): It is the **mean** difference P[o*] – P[o]

- E(S/M) : Input instances from D_x
- P(S/M) : Input instances from P_x
- N(S/M) : Input instances from O_x

- GE: Fluency Degradations
- RS: Semantic Consistency

ROME: Evaluation

Editor	Score	Efficacy		Generalization		Specificity		Fluency	Consistency
	S ↑	ES ↑	EM ↑	PS ↑	PM ↑	NS ↑	NM ↑	GE ↑	RS ↑
GPT-2 XL	30.5	22.2 (0.9)	-4.8 (0.3)	24.7 (0.8)	-5.0 (0.3)	78.1 (0.6)	5.0 (0.2)	626.6 (0.3)	31.9 (0.2)
FT	65.1	100.0 (0.0)	98.8 (0.1)	87.9 (0.6)	46.6 (0.8)	**40.4 (0.7)**	**-6.2 (0.4)**	607.1 (1.1)	40.5 (0.3)
FT+L	66.9	99.1 (0.2)	91.5 (0.5)	**48.7 (1.0)**	28.9 (0.8)	70.3 (0.7)	3.5 (0.3)	621.4 (1.0)	37.4 (0.3)
KN	**35.6**	**28.7 (1.0)**	**-3.4 (0.3)**	**28.0 (0.9)**	**-3.3 (0.2)**	72.9 (0.7)	3.7 (0.2)	**570.4 (2.3)**	**30.3 (0.3)**
KE	52.2	84.3 (0.8)	33.9 (0.9)	75.4 (0.8)	14.6 (0.6)	**30.9 (0.7)**	**-11.0 (0.5)**	**586.6 (2.1)**	31.2 (0.3)
KE-CF	**18.1**	99.9 (0.1)	97.0 (0.2)	95.8 (0.4)	59.2 (0.8)	**6.9 (0.3)**	**-63.2 (0.7)**	**383.0 (4.1)**	**24.5 (0.4)**
MEND	57.9	99.1 (0.2)	70.9 (0.8)	65.4 (0.9)	12.2 (0.6)	**37.9 (0.7)**	**-11.6 (0.5)**	**624.2 (0.4)**	34.8 (0.3)
MEND-CF	**14.9**	**100.0 (0.0)**	**99.2 (0.1)**	**97.0 (0.3)**	**65.6 (0.7)**	**5.5 (0.3)**	**-69.9 (0.6)**	**570.0 (2.1)**	33.2 (0.3)
ROME	**89.2**	100.0 (0.1)	97.9 (0.2)	96.4 (0.3)	62.7 (0.8)	**75.4 (0.7)**	**4.2 (0.2)**	621.9 (0.5)	**41.9 (0.3)**
GPT-J	23.6	16.3 (1.6)	-7.2 (0.7)	18.6 (1.5)	-7.4 (0.6)	83.0 (1.1)	7.3 (0.5)	621.8 (0.6)	29.8 (0.5)
FT	**25.5**	**100.0 (0.0)**	**99.9 (0.0)**	96.6 (0.6)	71.0 (1.5)	**10.3 (0.8)**	**-50.7 (1.3)**	**387.8 (7.3)**	**24.6 (0.8)**
FT+L	68.7	99.6 (0.3)	95.0 (0.6)	**47.9 (1.9)**	30.4 (1.5)	78.6 (1.2)	**6.8 (0.5)**	**622.8 (0.6)**	35.5 (0.5)
MEND	63.2	97.4 (0.7)	71.5 (1.6)	**53.6 (1.9)**	11.0 (1.3)	53.9 (1.4)	**-6.0 (0.9)**	620.5 (0.7)	32.6 (0.5)
ROME	**91.5**	99.9 (0.1)	99.4 (0.3)	**99.1 (0.3)**	**74.1 (1.3)**	**78.9 (1.2)**	5.2 (0.5)	620.1 (0.9)	**43.0 (0.6)**

Knowledge Editing for Large Language Models: A Survey

Song Wang, Yaochen Zhu, Haochen Liu, Zaiyi Zheng, Chen Chen, Jundong Li

Large language models (LLMs) have recently transformed both the academic and industrial landscapes due to their remarkable capacity to understand, analyze, and generate texts based on their vast knowledge and reasoning ability. Nevertheless, one major drawback of LLMs is their substantial computational cost for pre-training due to their unprecedented amounts of parameters. The disadvantage is exacerbated when new knowledge frequently needs to be introduced into the pre-trained model. Therefore, it is imperative to develop effective and efficient techniques to update pre-trained LLMs. Traditional methods encode new knowledge in pre-trained LLMs through direct fine-tuning. However, naively re-training LLMs can be computationally intensive and risks degenerating valuable pre-trained knowledge irrelevant to the update in the model. Recently, Knowledge-based Model Editing (KME) has attracted increasing attention, which aims to precisely modify the LLMs to incorporate specific knowledge, without negatively influencing other irrelevant knowledge. In this survey, we aim to provide a comprehensive and in-depth overview of recent advances in the field of KME. We first introduce a general formulation of KME to encompass different KME strategies. Afterward, we provide an innovative taxonomy of KME techniques based on how the new knowledge is introduced into pre-trained LLMs, and investigate existing KME strategies while analyzing key insights, advantages, and limitations of methods from each category. Moreover, representative metrics, datasets, and applications of KME are introduced accordingly. Finally, we provide an in-depth analysis regarding the practicality and remaining challenges of KME and suggest promising research directions for further advancement in this field.

Comments: Accepted by ACM Computing Surveys
Subjects: **Computation and Language (cs.CL)**; Artificial Intelligence (cs.AI)
Cite as: arXiv:2310.16218 **[cs.CL]**
(or arXiv:2310.16218v4 **[cs.CL]** for this version)
https://doi.org/10.48550/arXiv.2310.16218

https://arxiv.org/abs/2310.16218