Parametric Knowledge in Language Models

Sikun Guo (qkm6sq), Haochen Liu (sat2pv), Rui Zhao (dkw7xn)
Feb. 12, 2024

Language Models as Knowledge Bases?

Fabio Petroni, Tim Rocktaschel, Patrick Lewis, Anton Bakhtin,
Yuxiang Wu, Alexander H. Miller, Sebastian Riedel,
Facebook AI Research
University College London

Outline

1. Motivation and Background
2. Methodology and Design
3. Results
4. Conclusion and Limitations

Motivation

Language Model

- No schema engineering required
- Allow querying about an open class of relations
- Easy to extend to more data
- Could be pre-trained on massive data without requiring human supervision.

VS

Structured Knowledge Bases

- Schema engineering required
- Fixed relation and limited resources
- Require human to extend to more data

Motivation

Knowledge Bases: requires complex NLP pipelines involving entity extraction, coreference resolution, entity linking and relation extraction.

Neural Language Model: Just ask: “**Dante was born in [Mask]”**

Motivation

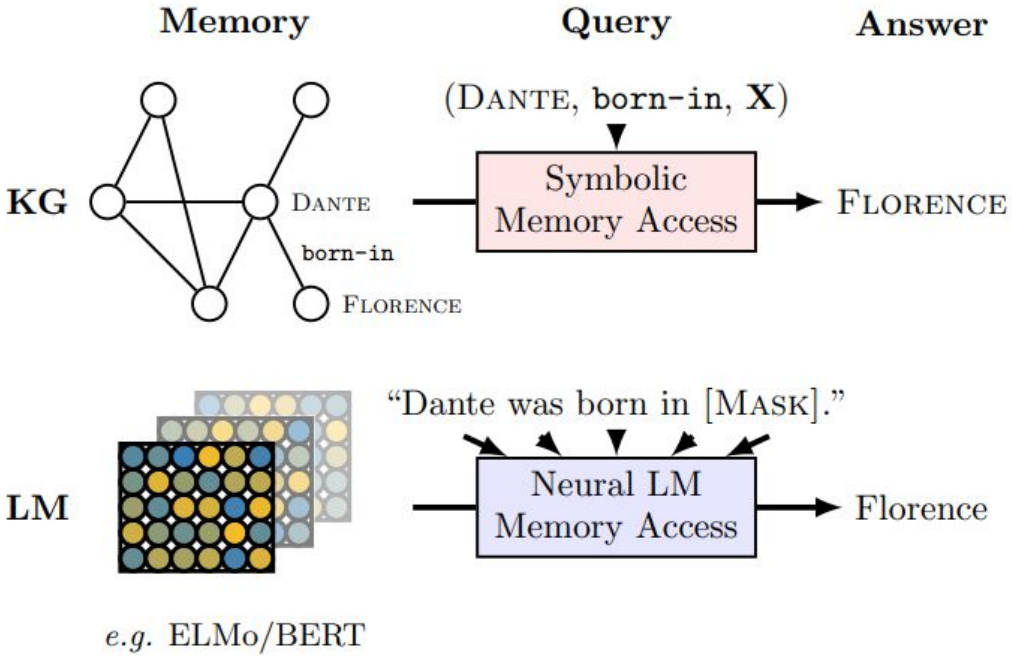
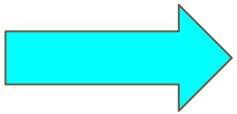
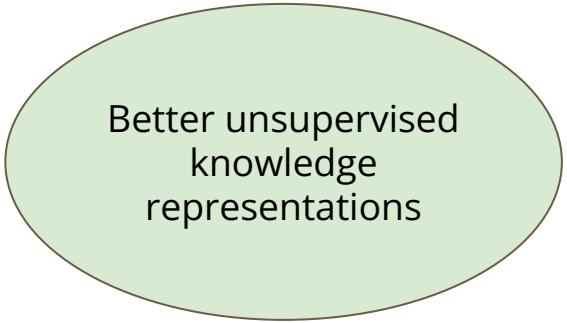


Figure Source: https://arxiv.org/pdf/1909.01066.pdf

Motivation

- How much relational knowledge do they **store**?
- How does this differ for different types of knowledge such as **facts about entities**, **common sense**, and **general question answering**?
- How does their performance **without fine-tuning** compare to symbolic knowledge bases automatically extracted from text?





Baseline

Model	Base Model	#Parameters	Training Corpus	Corpus Size
fairseq-fconv (Dauphin et al., 2017)	ConvNet	324M	WikiText-103	103M Words
Transformer-XL (large) (Dai et al., 2019)	Transformer	257M	WikiText-103	103M Words
ELMo (original) (Peters et al., 2018a)	BiLSTM	93.6M	Google Billion Word	800M Words
ELMo 5.5B (Peters et al., 2018a)	BiLSTM	93.6M	Wikipedia (en) & WMT 2008-2012	5.5B Words
BERT (base) (Devlin et al., 2018a)	Transformer	110M	Wikipedia (en) & BookCorpus	3.3B Words
BERT (large) (Devlin et al., 2018a)	Transformer	340M	Wikipedia (en) & BookCorpus	3.3B Words

Figure Source: https://arxiv.org/pdf/1909.01066.pdf

Baseline

- Freq: For a subject and relation pair, this baseline ranks words based on how frequently they appear as objects for the given relation in the test data.

- RE: For the relation-based knowledge sources, this paper used the pretrained Relation Extraction (RE) model [1]. It extracts relation triples from a given sentence using an LSTM-based encoder and an attention mechanism.

- DrQA: DrQA predicts answers to natural language questions using a two step pipeline. First, a TF/IDF information retrieval step is used to find relevant articles from a large store of documents (e.g. Wikipedia). On the retrieved top k articles, a neural reading comprehension model then extracts answers.

[1]: Daniil Sorokin and Iryna Gurevych. 2017. Contextaware representations for knowledge base relation extraction. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017, pages 1784–1789.

Metrics

- **Rank-based metrics and compute results per relation along with mean values across all relations.**

- **Use the mean precision at k (P@k). For a given fact, this value is 1 if the object is ranked among the top k results, and 0 otherwise.**

LAMA (LAnguage Model Analysis) Probe

- **Evaluate each model based on how highly it ranks the ground truth token against every other word in a fixed candidate vocabulary.**

- **Models which rank ground truth tokens high for these cloze statements have more factual knowledge**

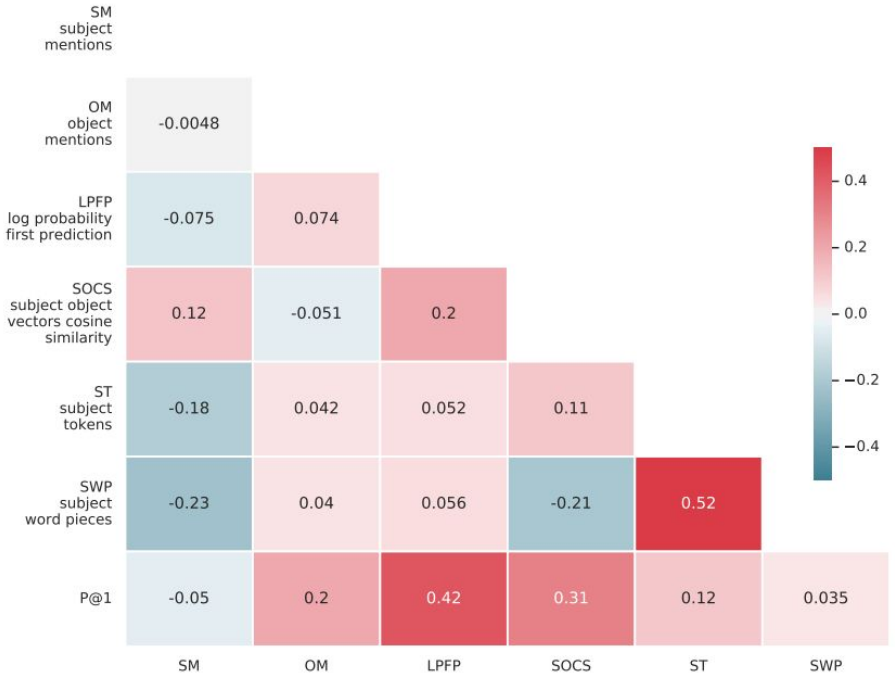
Results

Corpus	Relation	Statistics		Baselines		KB		LM					
		#Facts	#Rel	Freq	DrQA	RE_n	RE_o	Fs	Txl	Eb	E5B	Bb	Bl
Google-RE	birth-place	2937	1	4.6	-	3.5	13.8	4.4	2.7	5.5	7.5	14.9	**16.1**
	birth-date	1825	1	1.9	-	0.0	**1.9**	0.3	1.1	0.1	0.1	1.5	1.4
	death-place	765	1	6.8	-	0.1	7.2	3.0	0.9	0.3	1.3	13.1	**14.0**
	Total	5527	3	4.4	-	1.2	7.6	2.6	1.6	2.0	3.0	9.8	**10.5**
T-REx	1-1	937	2	1.78	-	0.6	10.0	17.0	36.5	10.1	13.1	68.0	**74.5**
	N-1	20006	23	23.85	-	5.4	**33.8**	6.1	18.0	3.6	6.5	32.4	34.2
	N-*M*	13096	16	21.95	-	7.7	**36.7**	12.0	16.5	5.7	7.4	24.7	24.3
	Total	34039	41	22.03	-	6.1	**33.8**	8.9	18.3	4.7	7.1	31.1	32.3
ConceptNet	Total	11458	16	4.8	-	-	-	3.6	5.7	6.1	6.2	15.6	**19.2**
SQuAD	Total	305	-	-	**37.5**	-	-	3.6	3.9	1.6	4.3	14.1	17.4

Figure Source: https://arxiv.org/pdf/1909.01066.pdf

Results

- SM and OM refer to the number of times a subject and an object are mentioned in the BERT training corpus respectively.
- LPFP is the log probability score associated with the first prediction.
- SOCS is the cosine similarity between subject and object vectors.
- ST and SWP are the number of tokens in the subject with a standard tokenization and the BERT WordPiece tokenization respectively.



Conclusion

- Presented a systematic analysis of the factual and commonsense knowledge in publicly available pretrained language models.
- BERT-large is able to recall such knowledge better than its competitors and at a level remarkably competitive with non-neural and supervised alternatives.
- It is non-trivial to extract a knowledge base from text that performs on par to directly using pretrained BERT-large.
- Relation extraction performance might be difficult to improve with more data. (Wikitext-103)

Limitations

- What if the knowledge are **fake**?

  The knowledge extracted from LMs are not guaranteed to be correct！

- How about the generally **cost**?

  The size of LMs matters for the performance of knowledge retrieval, and it's generally more expensive to serve/deploy a large LM than a knowledge base.

How Much Knowledge Can You Pack Into the Parameters of a Language Model?

Adam Roberts, Colin Raffel, Noam Shazeer
Google

Outline

1. Motivation and Background
2. Methodology and Design
3. Results
4. Conclusion and Limitations

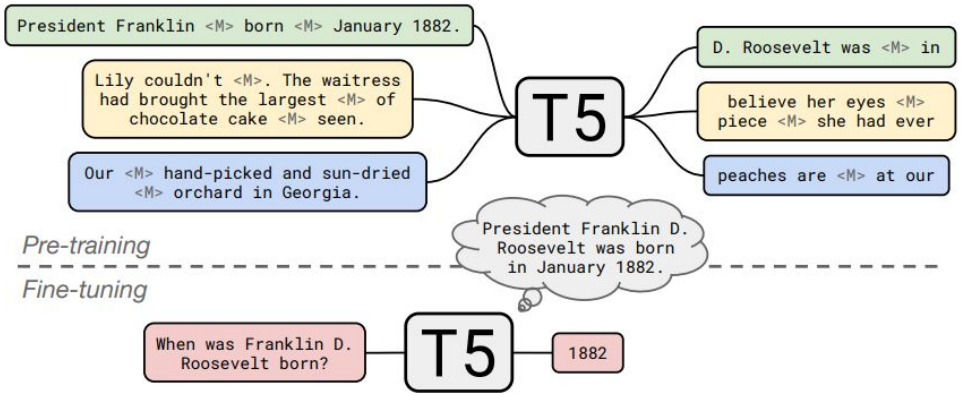
Motivation

Focusing on Model itself:

How Much Knowledge Can You Pack Into the Parameters of a Language Model?

—Fine-tune the model to answer questions without access to any external knowledge or context

Methodology



This paper fine-tune T5 to answer questions without inputting any additional information or context. This forces T5 to answer questions based on “knowledge” that it internalized during pre-training.

Datasets

- ***Natural Questions***, a dataset of questions from web queries, each accompanied by a Wikipedia article containing the answer.

- ***WebQuestions***, comprising questions from web queries matched to corresponding entries in FreeBase.

- ***TriviaQA***, a collection of questions from quiz league websites where each question is accompanied by pages from web and Wikipedia searches that may contain the answer.

Training

- “Text-to-Text Transfer Transformer” (T5)
- This paper performed experiments with the Base (220 million parameters), Large (770 million), 3B (3 billion), and 11B (11 billion) variants of T5.
- For fine-tuning the T5 checkpoints: this paper follow the procedure used in Raffel et al. (2019)[1] without any additional hyperparameter tuning and use the AdaFactor optimizer (Shazeer and Stern, 2018)[2] with a constant learning rate of 0.001, 10% dropout rate, and a batch size of 196,608 tokens.

[1] Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2019. olmpics–on what language model pre-training captures. arXiv preprint arXiv:1912.13283.
[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in Neural Information Processing Systems.

Results

- Performance on each dataset improves as the model size increases.
- T5.1.1- XXL with SSM ultimately achieves state-of-the-art on WebQuestions.
- Most open-domain question answering systems must first do an expensive lookup step over the entire knowledge corpus and then attend to a long document to extract an answer. (T5 free)

Competitive on open-domain question answering!

	NQ	WQ	TQA	
			dev	test
Chen et al. (2017)	–	20.7	–	–
Lee et al. (2019)	33.3	36.4	47.1	–
Min et al. (2019a)	28.1	–	50.9	–
Min et al. (2019b)	31.8	31.6	55.4	–
Asai et al. (2019)	32.6	–	–	–
Ling et al. (2020)	–	–	35.7	–
Guu et al. (2020)	40.4	40.7	–	–
Févry et al. (2020)	–	–	43.2	53.4
Karpukhin et al. (2020)	**41.5**	42.4	**57.9**	–
T5-Base	25.9	27.9	23.8	29.1
T5-Large	28.5	30.6	28.7	35.9
T5-3B	30.4	33.6	35.1	43.4
T5-11B	32.6	37.2	42.3	50.1
T5-11B + SSM	34.8	40.8	51.0	60.5
T5.1.1-Base	25.7	28.2	24.2	30.6
T5.1.1-Large	27.3	29.5	28.5	37.2
T5.1.1-XL	29.5	32.4	36.0	45.1
T5.1.1-XXL	32.8	35.6	42.9	52.5
T5.1.1-XXL + SSM	35.2	**42.8**	51.9	**61.6**

Interesting reasons

- Answers with meaning-preserving differences in phrasing (e.g. “April 15” vs. “April 15th”).
- Questions that were missing all possible correct answers.
- Some questions were unanswerable without knowing the exact time or article they referred to (e.g. “what is the latest version of microsoft office 2010” depends on when the question is being asked).

So : underestimated

Conclusion

Large language models pre-trained on **unstructured** text can attain competitive results on open-domain question answering benchmarks **without** any access to external knowledge. (11 billion parameters)

Limitations

- **Underestimated: We need better metrics.**
- **Human evaluation is hard.**

Transformer Feed-Forward Layers Are Key-Value Memories

Mor Geva, Roei Schuster, Jonathan Berant, Omer Levy
Blavatnik School of Computer Science, Tel-Aviv University
Allen Institute for Artificial Intelligence
Cornell Tech

Outline

1. Introduction: Feed-Forward Layers as Key-Value Memories
2. Keys Capture Input Patterns
3. Values Represent Distributions
4. Aggregating Memories

Feed-Forward Layers as Key-Value Memories

Feed-forward layers: $8d^2$

self-attention layers: $4d^2$

d is the model's hidden dimension

Feed-forward layers emulate neural memories

Parameters in the model: key and value

Each key correlates with a specific set of human-interpretable input patterns!

Feed-forward Layers

A position-wise function, processing each input vector independently.

$$\mathrm{FF}(\mathbf{x}) = f(\mathbf{x} \cdot K^\top) \cdot V \qquad \mathbf{x} \in \mathbb{R}^d \qquad K, V \in \mathbb{R}^{d_m \times d}$$

f is a non-linearity such as **ReLU**.

Neural memory

A neural memory consists of d_m key-value pairs.

For $\mathbf{x} \in \mathbb{R}^d$

Each key: $\mathbf{k}_i \in \mathbb{R}^d$

Together form: $K \in \mathbb{R}^{d_m \times d}$

Value: $V \in \mathbb{R}^{d_m \times d}$

Neural memory

A **distribution** over the keys:

$$p(k_i \mid x) \propto \exp(\mathbf{x} \cdot \mathbf{k}_i)$$

$$\mathrm{MN}(\mathbf{x}) = \sum_{i=1}^{d_m} p(k_i \mid x)\mathbf{v}_i$$

In the matrix:

$$\mathrm{MN}(\mathbf{x}) = \mathrm{softmax}(\mathbf{x} \cdot K^\top) \cdot V$$

Feed-forward Layers Emulate Neural Memory

$$\mathrm{FF}(\mathbf{x}) = f(\mathbf{x} \cdot K^\top) \cdot V$$

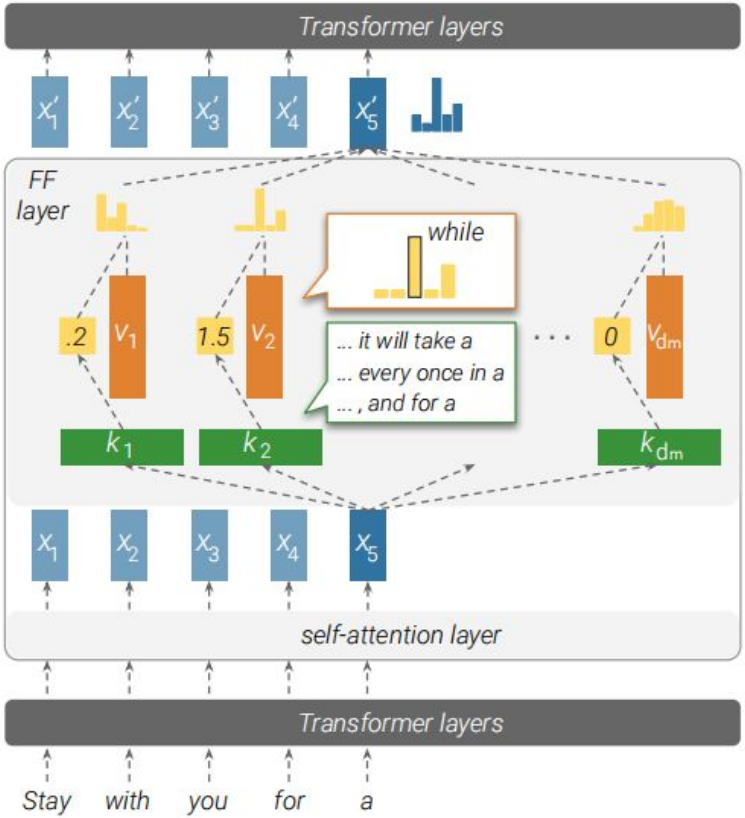
$$\mathrm{MN}(\mathbf{x}) = \mathrm{softmax}(\mathbf{x} \cdot K^\top) \cdot V$$

Difference: transformer does not use a normalizing function.

hidden dimension d_m: the number of memories in the layer.

hidden layer $f(\mathbf{x} \cdot K^\top)$: a vector containing an unnormalized non-negative coefficient for each memory.

Feed-Forward Layers as Key-Value Memories



Keys Capture Input Patterns

Each individual key vector in **K** corresponds to a specific pattern over the input prefix.

Test:

Retrieve the training examples.

most associated with a given key = the input texts where the memory coefficient is highest

Experiment

LM: a 16-layer transformer language model trained on WikiText-103.

$d = 1024$, $d_m = 4096$

Randomly sample 10 keys per layer.

Retrieving trigger examples:

Given a key k_i^l compute $\text{ReLU}(\mathbf{x}_j^\ell \cdot \mathbf{k}_i^\ell)$ for every prefix.

Retrieve the top-t trigger examples.

Pattern analysis

Key	Pattern	Example trigger prefixes
k^{1}_{449}	Ends with *"substitutes"* (shallow)	*At the meeting, Elton said that "for artistic reasons there could be no substitutes* *In German service, they were used as substitutes* *Two weeks later, he came off the substitutes*
k^{6}_{2546}	Military, ends with *"base"/"bases"* (shallow + semantic)	*On 1 April the SRSG authorised the SADF to leave their bases* *Aircraft from all four carriers attacked the Australian base* *Bombers flying missions to Rabaul and other Japanese bases*
k^{10}_{2997}	a "part of" relation (semantic)	*In June 2012 she was named as one of the team that competed* *He was also a part of the Indian delegation* *Toy Story is also among the top ten in the BFI list of the 50 films you should*
k^{13}_{2989}	Ends with a time range (semantic)	*Worldwide, most tornadoes occur in the late afternoon, between 3 pm and 7* *Weekend tolls are in effect from 7:00 pm Friday until* *The building is open to the public seven days a week, from 11:00 am to*
k^{16}_{1935}	TV shows (semantic)	*Time shifting viewing added 57 percent to the episode's* *The first season set that the episode was included in was as part of the* *From the original NBC daytime version , archived*

Table 1: Examples of human-identified patterns that trigger different memory keys.

Results

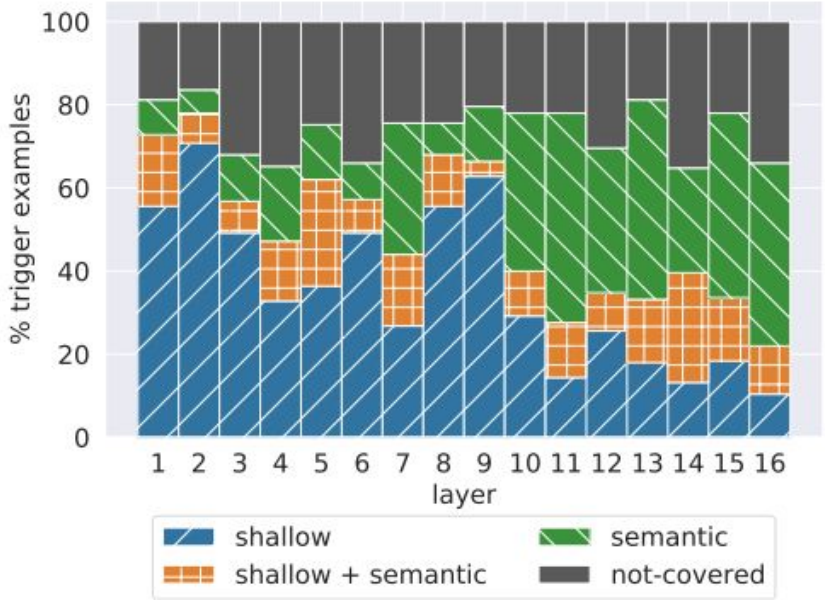


Figure 2: Breakdown of the labels experts assigned to trigger examples in each layer. Some examples were not associated with any pattern ("not-covered").

At least one pattern for every key.

Average: 3.6

Shallow layers detect shallow patterns

Layers in deep contextualized models encode shallow (semantic) features of the inputs.

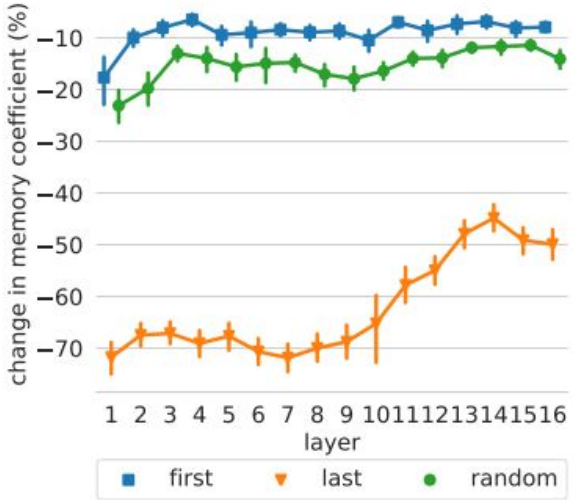


Figure 3: Relative change in memory coefficient caused by removing the first, the last, or a random token from the input.

Values Represent Distributions

Each value can be viewed as a distribution over the output vocabulary, and demonstrate that this distribution complements the patterns in the corresponding key in the model's upper layers.

Casting values as distributions over the vocabulary.

$$\mathbf{p}_i^\ell = \text{softmax}(\mathbf{v}_i^\ell \cdot E).$$

Values Represent Distributions

Agreement rate: the fraction of memory cells (dimensions) where the value's top prediction matches the key's top trigger example.

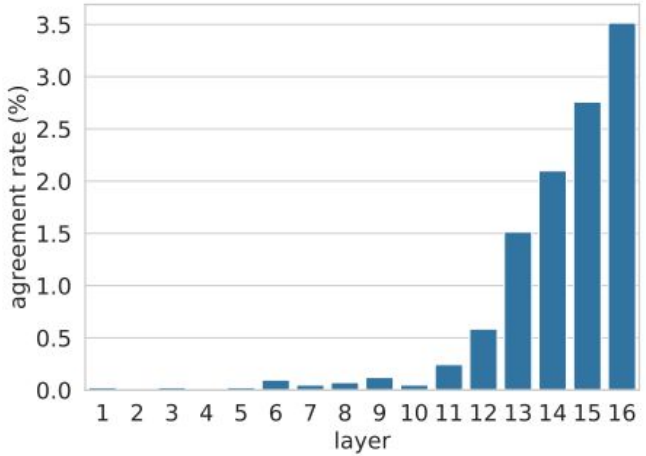


Figure 4: Agreement rate between the top-ranked token based on the value vector $\mathbf{v}_i^\ell$, and the next token of the top-ranked trigger example associated with the key vector $\mathbf{k}_i^\ell$.

Values Represent Distributions

Where the key's top-1 trigger example ranks in the value vector's distribution?

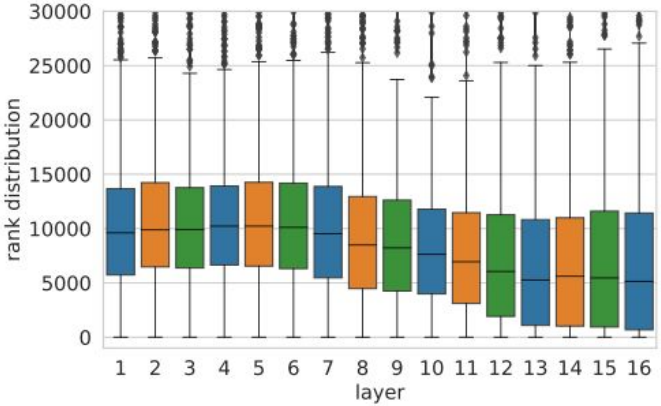


Figure 5: Distribution of the rank of the next-token in the top-1 trigger example of $\mathbf{k}_i^\ell$ (w_i^ℓ), according to the ranking induced by the value vector $\mathbf{v}_i^\ell$. We cut the tail of the distribution, which stretches up to the vocabulary size (~270K tokens).

Values Represent Distributions

The probability of the values' top prediction

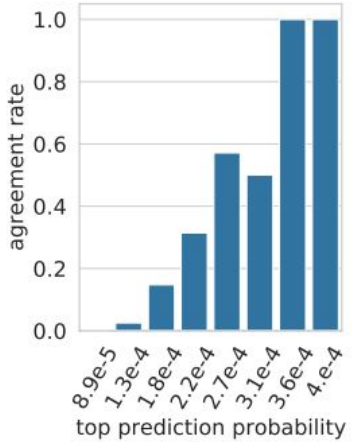


Figure 6: Agreement rate (between the top-ranked token based on the value vector $\mathbf{v}_i^\ell$ and the next token of the top-ranked trigger example associated with the key vector $\mathbf{k}_i^\ell$) as a function of the maximal probability assigned by the value vector.

Aggregating Memories

Information from multiple cells and layers ⇨ a model-wide prediction

Every feed-forward layer combines multiple memories to produce a distribution that is qualitatively different from each of its component memories' value distributions.

Different layers: combined via residual connections in a refinement process

Intra-Layer Memory Composition

The feed-forward layer's output can be defined as the sum of value vectors weighted by their memory coefficients:

$$\mathbf{y}^\ell = \sum_i \mathrm{ReLU}(\mathbf{x}^\ell \cdot \mathbf{k}_i^\ell) \cdot \mathbf{v}_i^\ell + \mathbf{b}^\ell.$$

First measure the fraction of "active" memories (cells with a non-zero coefficient)

Intra-Layer Memory Composition

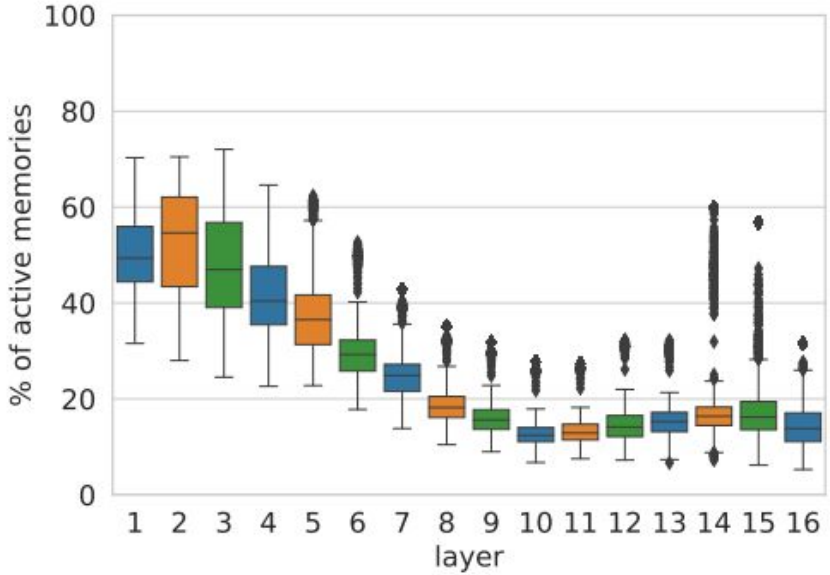


Figure 7: The fraction of active memories (i.e., with positive memory coefficient) out of 4096 memories in every layer, for a random sample of 4,000 examples.

Intra-Layer Memory Composition

Count the number of instances where the feed-forward layer's top prediction is different from all of the memories' top predictions.

$$\text{top}(\mathbf{h}) = \text{argmax}(\mathbf{h} \cdot E)$$

$$\forall i : \text{top}(\mathbf{v}_i^\ell) \neq \text{top}(\mathbf{y}^\ell)$$

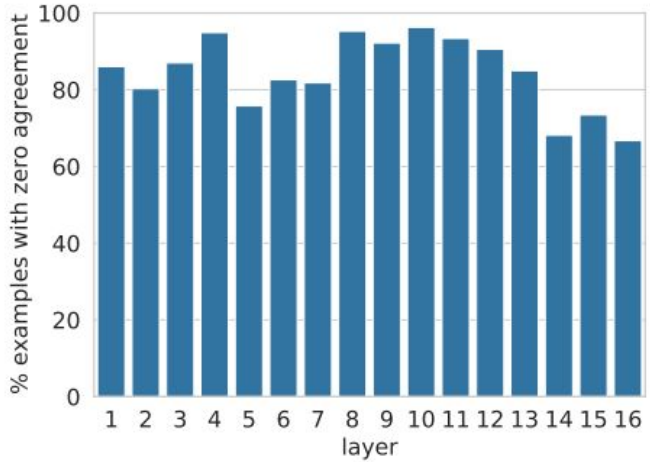


Figure 8: The fraction of examples in a random sample of 4,000 examples where the layer's prediction is different from the prediction of all of its memories.

Inter-Layer Prediction Refinement

The model uses the sequential composition apparatus as a means to refine its prediction from layer to layer, often deciding what the prediction will be at one of the lower layers.

To test: if the probability distribution induced by the residual vector matches the model's final output

Inter-Layer Prediction Refinement

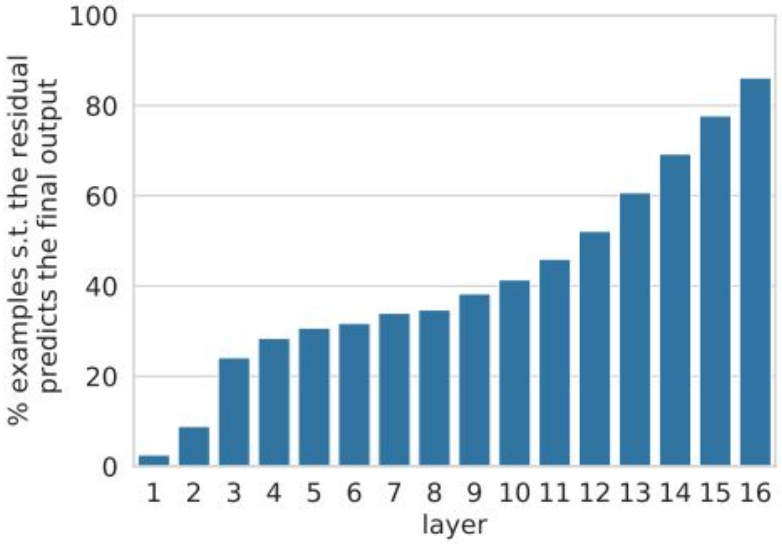


Figure 9: Fraction of examples in each layer, where the residual's top prediction matches the model's output.

Inter-Layer Prediction Refinement

The probability mass that each layer's residual vector assigns to the model's final prediction:

$$w = \text{top}(\mathbf{o}^L)$$
$$\mathbf{p} = \text{softmax}(\mathbf{r}^\ell \cdot E)$$
$$p = \mathbf{p}_w$$

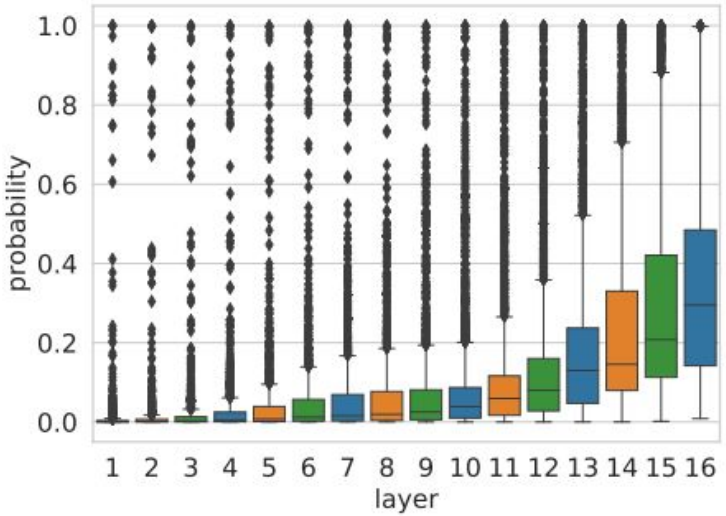


Figure 10: Probability of the token output by the model according to the residual of each layer.

Conclusion

(a) Keys are correlated with human-interpretable input patterns

(b) Values, mostly in the model's upper layers, induce distributions over the output vocabulary that correlate with the next-token distribution of patterns in the corresponding key

(c) The model's output is formed via an aggregation of the distributions, whereby they are first composed to form individual layer outputs, which are then refined throughout the model's layers using residual connections.

Limitation

1. The human-identifiable knowledge/patterns are distributed in the model rather arbitrarily.
2. No guarantee that every feedforward layer stores knowledge.

Locating and Editing Factual Associations in GPT

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

Motivations

The authors are interested in **how** and **where** a model stores its factual associations for 2 reasons:

1. To better understand huge opaque neural networks.
2. Fixing mistakes.

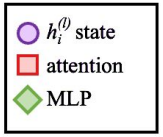
How to formulate factual associations?

Authors use **knowledge tuples t=(s,r,o)**

Example: **S** = Eiffel Tower, **r** = is located in the city of, **o** = Paris

To query GPT for knowledge of a fact, the authors express **(s, r)** as a text prompt (by expanding a template from the **CounterFact** data set), and check whether the generated continuation matches **o**.

Locating Factual Associations in GPT: Causal Tracing



Overview of Causal Tracing for embeddings:

Corrupt, restore, and compare the differences after 3 runs.

$$\mathbb{P}[o] \quad \mathbb{P}_*[o] \quad \mathbb{P}_{*,\,\text{clean}\ h_i^{(l)}}[o]$$

Total Effect:

$$\text{TE} = \mathbb{P}[o] - \mathbb{P}_*[o]$$

Indirect Effect

$$\text{IE} = \mathbb{P}_{*,\,\text{clean}\ h_i^{(l)}}[o] - \mathbb{P}_*[o]$$

Locating Factual Associations in GPT: Causal Tracing

Results of Causal Tracing for embeddings— Average Indirect Effect:

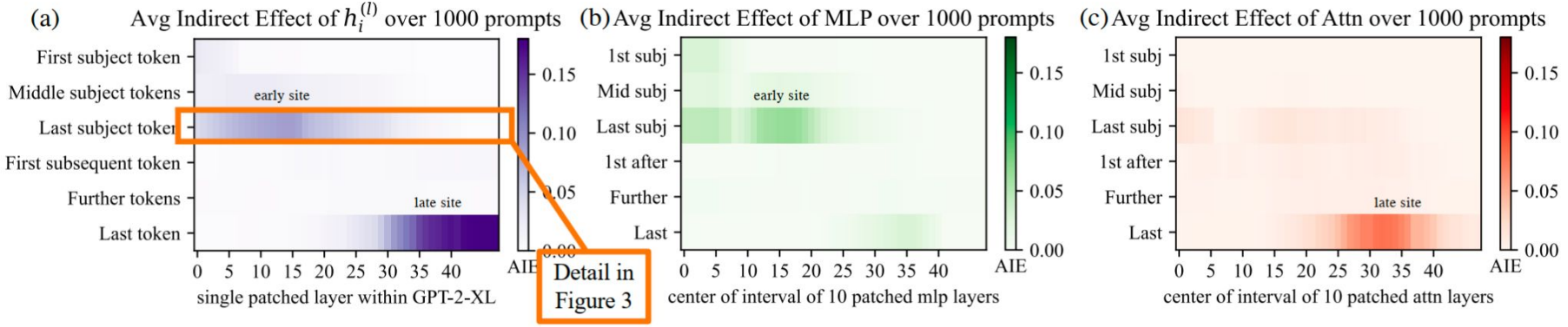


Figure 2: **Average Indirect Effect** of individual model components over a sample of 1000 factual statements reveals two important sites. (a) Strong causality at a 'late site' in the last layers at the last token is unsurprising, but strongly causal states at an 'early site' in middle layers at the last subject token is a new discovery. (b) MLP contributions dominate the early site. (c) Attention is important at the late site. Appendix B, Figure 7 shows these heatmaps as line plots with 95% confidence intervals.

Locating Factual Associations in GPT: Causal Tracing

Results of Causal Tracing for embeddings—Causal effects with a modified computation graph:

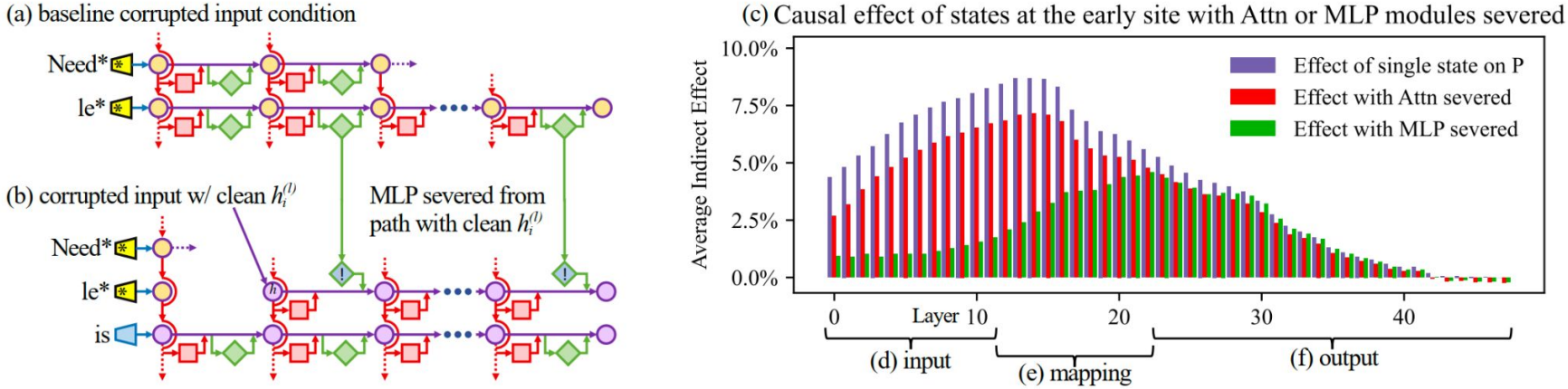


Figure 3: **Causal effects with a modified computation graph**. (a,b) To isolate the effects of MLP modules when measuring causal effects, the computation graph is modified. (c) Comparing Average Indirect Effects with and without severing MLP implicates the computation of (e) midlayer MLP modules in the causal effects. No similar gap is seen when attention is similarly severed.

Editing Factual Associations in GPT: Rank-One Model Editing

The Localized Factual Association Hypothesis:

1. Each midlayer MLP module accepts inputs that encode a subject, then produces outputs that recall memorized properties about that subject.

2. Middle layer MLP outputs accumulate information, then the summed information is copied to the last token by attention at high layers.

Editing Factual Associations in GPT: Rank-One Model Editing

Rank-One Model Editing (ROME) Overview:

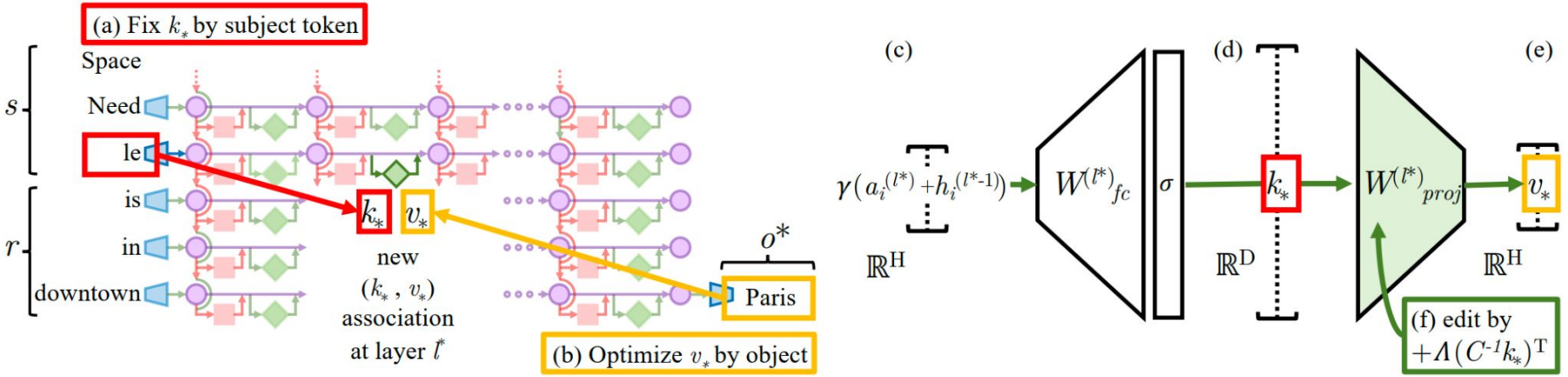


Figure 4: **Editing one MLP layer with ROME**. To associate *Space Needle* with *Paris*, the ROME method inserts a new (k_*, v_*) association into layer l^*, where (a) key k_* is determined by the subject and (b) value v_* is optimized to select the object. (c) Hidden state at layer l^* and token i is expanded to produce (d) the key vector k_* for the subject. (e) To write new value vector v_* into the layer, (f) we calculate a rank-one update $\Lambda(C^{-1}k_*)^T$ to cause $\hat{W}^{(l)}_{proj}k_* = v_*$ while minimizing interference with other memories stored in the layer.

Editing Factual Associations in GPT: Rank-One Model Editing

Rank-One Model Editing logic behind: Viewing the Transformer MLP as an Associative Memory

$$WK \approx V$$

$$W = VK^{+}$$

minimize $\|\hat{W}K - V\|$ such that $\hat{W}k_* = v_*$ by setting $\hat{W} = W + \Lambda(C^{-1}k_*)^T$

Editing Factual Associations in GPT: Rank-One Model Editing

Rank-One Model Editing Step by Step:

Step 1: Choosing k∗ to Select the Subject

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

Editing Factual Associations in GPT: Rank-One Model Editing

Rank-One Model Editing Step by Step:

Step 2: Choosing v∗ to Recall the Fact

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

Editing Factual Associations in GPT: Rank-One Model Editing

Rank-One Model Editing Step by Step:

Step 3: Inserting the Fact using rank-one update

$$\text{minimize } \|\hat{W}K - V\| \text{ such that } \hat{W}k_* = v_* \quad \text{by setting } \hat{W} = W + \Lambda(C^{-1}k_*)^T$$

Editing Factual Associations in GPT: Rank-One Model Editing

Rank-One Model Editing evaluation: Distinguish Knowing a Fact from Saying a Fact

Specificity: when your knowledge of a fact changes, it doesn't change other facts.

Generalization: knowledge of a fact is robust to changes in wording and context.

The authors did a set of experiments to verify the effectiveness of ROME regarding the 2 major concerns mentioned above.

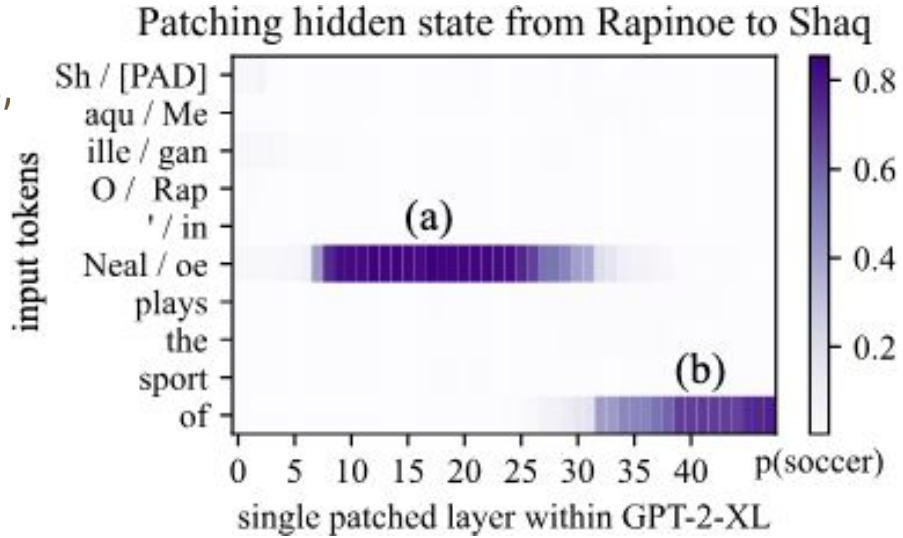
Major Contributions

1. **Factual associations can be localized** along three dimensions, to

(1) MLP module parameters

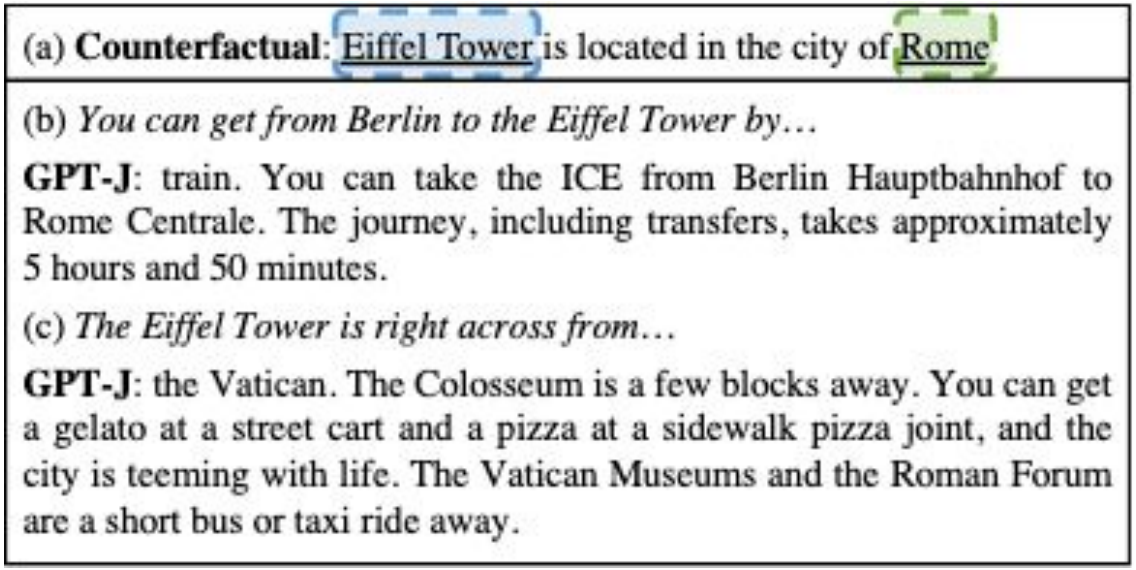
(2) at a range of middle layers

(3) specifically during processing of the last token of the subject.



Major Contributions

2. **Individual factual associations can be changed** by making small rank-one changes in a single MLP module.



Limitations

1. ROME edits a single fact at a time, and it is not intended as a practical method for large-scale model training.
2. ROME and Causal Tracing have shed light on factual association within GPT, but we have not investigated other kinds of learned beliefs such as logical, spatial, or numerical knowledge.
3. The understanding of the structure of the vector spaces that represent learned attributes remains incomplete.

Recap of this lecture: from language to knowledge

1. Besides processing language, the large training corpus enables Language Models to store relational knowledge too.
2. The knowledge can be located in Language Models parameters, where feed-forward layers can be treated as key-value memories.
3. The knowledge of factual associations in Language Models can be changed via rank-one update.