Locating and Editing Factual Associations in GPT

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

Abstract

We analyze the storage and recall of factual associations in autoregressive transformer language models, finding evidence that these associations correspond to localized, directly-editable computations. We first develop a causal intervention for identifying neuron *activations* that are decisive in a model's factual predictions. This reveals a distinct set of steps in middle-layer feed-forward modules that mediate factual predictions while processing subject tokens. To test our hypothesis that these computations correspond to factual association recall, we modify feed-forward *weights* to update specific factual associations using Rank-One Model Editing (ROME). We find that ROME is effective on a standard zero-shot relation extraction (zsRE) model-editing task. We also evaluate ROME on a new dataset of difficult counterfactual assertions, on which it simultaneously maintains both specificity and generalization, whereas other methods sacrifice one or another. Our results confirm an important role for mid-layer feed-forward modules in storing factual associations and suggest that direct manipulation of computational mechanisms may be a feasible approach for model editing. The code, dataset, visualizations, and an interactive demo notebook are available at https://rome.baulab.info/.

1 Introduction

Where does a large language model store its facts? In this paper, we report evidence that factual associations in GPT correspond to a localized computation that can be directly edited.

Large language models can predict factual statements about the world (Petroni et al., 2019; Jiang et al., 2020; Roberts et al., 2020). For example, given the prefix "*The Space Needle is located in the city of*," GPT will reliably predict the true answer: "*Seattle*" (Figure 1a). Factual knowledge has been observed to emerge in both autoregressive GPT models (Radford et al., 2019; Brown et al., 2020) and masked BERT models (Devlin et al., 2019).

In this paper, we investigate how such factual associations are stored within GPT-like autoregressive transformer models. Although many of the largest neural networks in use today are autoregressive, the way that they store knowledge remains under-explored. Some research has been done for masked models (Petroni et al., 2019; Jiang et al., 2020; Elazar et al., 2021a; Geva et al., 2021; Dai et al., 2022; De Cao et al., 2021), but GPT has architectural differences such as unidirectional attention and generation capabilities that provide an opportunity for new insights.

We use two approaches. First, we trace the causal effects of hidden state activations within GPT using causal mediation analysis (Pearl, 2001; Vig et al., 2020b) to identify the specific modules that mediate recall of a fact about a subject (Figure 1). Our analysis reveals that feedforward MLPs at a range of middle layers are decisive when processing the last token of the subject name (Figures 1b,2b,3).

Second, we test this finding in model weights by introducing a Rank-One Model Editing method (ROME) to alter the parameters that determine a feedfoward layer's behavior at the decisive token.

*Equal contribution. Correspondence to mengk@mit.edu, davidbau@northeastern.edu.
†Supported by the Viterbi Fellowship in the Center for Computer Engineering at the Technion.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

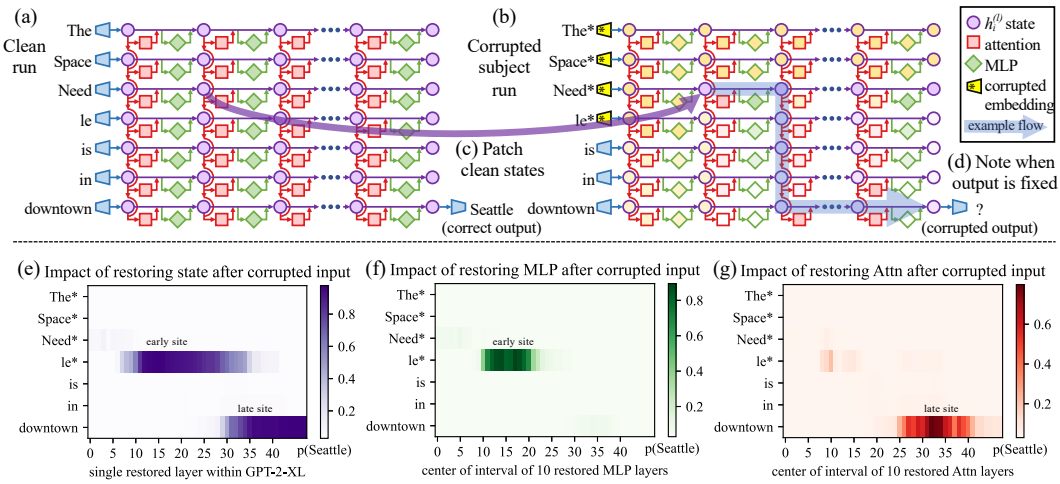

Figure 1: **Causal Traces** compute the causal effect of neuron activations by running the network twice: (a) once normally, and (b) once where we corrupt the subject token and then (c) restore selected internal activations to their clean value. (d) Some sets of activations cause the output to return to the original prediction; the light blue path shows an example of information flow. The causal impact on output probability is mapped for the effect of (e) each hidden state on the prediction, (f) only MLP activations, and (g) only attention activations.

Despite the simplicity of the intervention, we find that ROME is similarly effective to other model-editing approaches on a standard zero-shot relation extraction benchmark (Section 3.2).

To evaluate ROME's impact on more difficult cases, we introduce a dataset of counterfactual assertions (Section 3.3) that would not have been observed in pretraining. Our evaluations (Section 3.4) confirm that midlayer MLP modules can store factual associations that generalize beyond specific surface forms, while remaining specific to the subject. Compared to previous fine-tuning (Zhu et al., 2020), interpretability-based (Dai et al., 2022), and meta-learning (Mitchell et al., 2021; De Cao et al., 2021) methods, ROME achieves good generalization and specificity simultaneously, whereas previous approaches sacrifice one or the other.

2 Interventions on Activations for Tracing Information Flow

To locate facts within the parameters of a large pretrained autoregressive transformer, we begin by analyzing and identifying the specific hidden states that have the strongest causal effect on predictions of individual facts. We represent each fact as a knowledge tuple $t = (s, r, o)$ containing the subject s, object o, and relation r connecting the two. Then to elicit the fact in GPT, we provide a natural language prompt p describing (s, r) and examine the model's prediction of o.

An autoregressive transformer language model $G : \mathcal{X} \to \mathcal{Y}$ over vocabulary V maps a token sequence $x = [x_1, ..., x_T] \in \mathcal{X}$, $x_i \in V$ to a probability distribution $y \in \mathcal{Y} \subset \mathbb{R}^{|V|}$ that predicts next-token continuations of x. Within the transformer, the ith token is embedded as a series of hidden state vectors $h_i^{(l)}$, beginning with $h_i^{(0)} = \text{emb}(x_i) + \text{pos}(i) \in \mathbb{R}^H$. The final output $y = \text{decode}(h_T^{(L)})$ is read from the last hidden state.

We visualize the internal computation of G as a grid (Figure 1a) of hidden states $h_i^{(l)}$ in which each layer l (left $\to$ right) adds global attention $a_i^{(l)}$ and local MLP $m_i^{(l)}$ contributions computed from previous layers, and where each token i (top $\to$ bottom) attends to previous states from other tokens. Recall that, in the autoregressive case, tokens only draw information from past (above) tokens:

$$
\begin{aligned}
h_i^{(l)} &= h_i^{(l-1)} + a_i^{(l)} + m_i^{(l)} \\
a_i^{(l)} &= \text{attn}^{(l)}\left(h_1^{(l-1)}, h_2^{(l-1)}, \ldots, h_i^{(l-1)}\right) \\
m_i^{(l)} &= W_{proj}^{(l)}\, \sigma\left(W_{fc}^{(l)} \gamma\left(a_i^{(l)} + h_i^{(l-1)}\right)\right).
\end{aligned} \tag{1}
$$

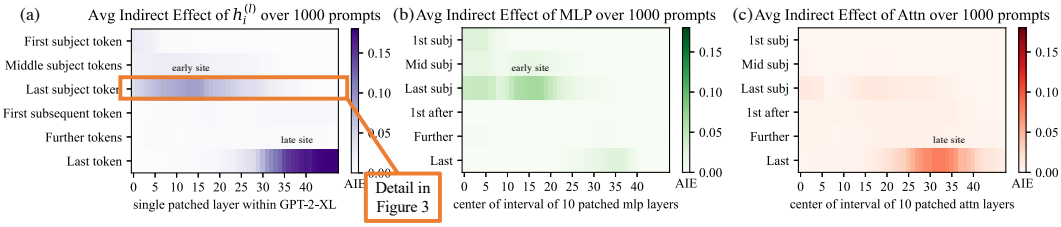

Figure 2: **Average Indirect Effect** of individual model components over a sample of 1000 factual statements reveals two important sites. (a) Strong causality at a 'late site' in the last layers at the last token is unsurprising, but strongly causal states at an 'early site' in middle layers at the 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.

Each layer's MLP is a two-layer neural network parameterized by matrices $W_{proj}^{(l)}$ and $W_{fc}^{(l)}$, with rectifying nonlinearity σ and normalizing nonlinearity γ. For further background on transformers, we refer to Vaswani et al. (2017).[3]

2.1 Causal Tracing of Factual Associations

The grid of states (Figure 1) forms a *causal graph* (Pearl, 2009) describing dependencies between the hidden variables. This graph contains many paths from inputs on the left to the output (next-word prediction) at the lower-right, and we wish to understand if there are specific hidden state variables that are more important than others when recalling a fact.

As Vig et al. (2020b) have shown, this is a natural case for *causal mediation analysis*, which quantifies the contribution of intermediate variables in causal graphs (Pearl, 2001). To calculate each state's contribution towards a correct factual prediction, we observe all of G's internal activations during three runs: a **clean** run that predicts the fact, a **corrupted** run where the prediction is damaged, and a **corrupted-with-restoration** run that tests the ability of a single state to restore the prediction.

- In the **clean run**, we pass a factual prompt x into G and collect all hidden activations $\{h_i^{(l)} \mid i \in [1,T], l \in [1,L]\}$. Figure 1a provides an example illustration with the prompt: "The Space Needle is in downtown _____", for which the expected completion is $o =$ "Seattle".
- In the baseline **corrupted run**, the subject is obfuscated from G before the network runs. Concretely, immediately after x is embedded as $[h_1^{(0)}, h_2^{(0)}, \ldots, h_T^{(0)}]$, we set $h_i^{(0)} := h_i^{(0)} + \epsilon$ for all indices i that correspond to the subject entity, where $\epsilon \sim \mathcal{N}(0; \nu)$[4]; . G is then allowed to continue normally, giving us a set of corrupted activations $\{h_{i*}^{(l)} \mid i \in [1,T], l \in [1,L]\}$. Because G loses some information about the subject, it will likely return an incorrect answer (Figure 1b).
- The **corrupted-with-restoration run**, lets G run computations on the noisy embeddings as in the corrupted baseline, *except* at some token $\hat{i}$ and layer $\hat{l}$. There, we hook G so that it is forced to output the clean state $h_{\hat{i}}^{(\hat{l})}$; future computations execute without further intervention. Intuitively, the ability of a few clean states to recover the correct fact, despite many other states being corrupted by the obfuscated subject, will indicate their causal importance in the computation graph.

Let $\mathbb{P}[o]$, $\mathbb{P}_*[o]$, and $\mathbb{P}_{*,\,\text{clean } h_i^{(l)}}[o]$ denote the probability of emitting o under the clean, corrupted, and corrupted-with-restoration runs, respectively; dependence on the input x is omitted for notational simplicity. The **total effect** (TE) is the difference between these quantities: $\text{TE} = \mathbb{P}[o] - \mathbb{P}_*[o]$. The **indirect effect** (IE) of a specific mediating state $h_i^{(l)}$ is defined as the difference between the probability of o under the corrupted version and the probability when that state is set to its clean version, while the subject remains corrupted: $\text{IE} = \mathbb{P}_{*,\,\text{clean } h_i^{(l)}}[o] - \mathbb{P}_*[o]$. Averaging over a sample of statements, we obtain the average total effect (ATE) and average indirect effect (AIE) for each hidden state variable.[5]

[3] Eqn. 1 calculates attention sequentially after the MLP module as in Brown et al. (2020). Our methods also apply to GPT variants such as Wang & Komatsuzaki (2021) that put attention in parallel to the MLP.

[4] We select ν to be 3 times larger than the empirical standard deviation of embeddings; see Appendix B.1 for details, and see Appendix B.4 for an analysis of other corruption rules.

[5] One could also compute the direct effect, which flows through other model components besides the chosen mediator. However, we found this effect to be noisy and uninformative, in line with results by Vig et al. (2020b).