Patching Language Models for Parallel Thinking

Shira Baneth and **Amnon Levi** and **Jonathan Yaffe**

shirabaneth@mail.tau.ac.il   amnonlevi@mail.tau.ac.il   jonathany@mail.tau.ac.il

Abstract

Examining the information encoded in the hidden representations of large language models can explain their behavior and verify their alignment with human values. Recent methods suggest using the model itself to elucidate its internal representations in natural language. A recent paper [Ghandeharioun et al., 2024] introduced a framework called Patchscopes, a framework that involves transferring specific information within the network. This paper demonstrated improvements by applying Patchscopes to multi-hop reasoning questions. In this study, we explored applying the Patchscopes method to different question structures. Our source code can be found on https://github.com/amnonlevitau/interpretability

1 Introduction

Multi-hop questions, exemplified with various structures in figure 1, require answering multiple sub-questions and then integrating their responses into diverse formats, posing a significant challenge for language models. By testing the transfer of information between different locations within a language model during its attempt to answer these questions, we can gain valuable insights into the model's functionality and improve its performance.

As introduced by Ghandeharioun et al. [2024], the central concept of Patchscopes leverages the advanced capabilities of large language models (LLMs) to generate human-like text, thereby translating the information encoded in their hidden representations. Specifically, given a hidden representation derived from an LLM inference pass, the Patchscopes framework decodes particular information by patching it into a different inference pass, facilitating the translation of that specific information.

The study by Ghandeharioun et al. [2024] demonstrated that passing information between different states in an LLM can significantly enhance

Figure 1: Question Structures from Trivedi et al. [2022]: The previous study [Ghandeharioun et al., 2024] primarily focused on the first type listed in this table, whereas the current study concentrates mainly on the third type.

performance on multi-hop questions, particularly those composed of 2-hop sub-questions. Their framework involves transferring information from the inference pass of one subquestion's answer to a different inference pass. In this paper, we investigate the impact of Patchscopes on comparison-based questions and have adapted the Patchscopes method to better accommodate this type of question.

2 Method

2.1 Original Patchscopes Framework

The original Patchscopes framework, as introduced in Ghandeharioun et al. [2024], involves decoding specific information from a representation within a Large Language Model (LLM) by "patching" it into the inference pass of a different prompt designed to facilitate the extraction of that information. We add here the formulation of the Patchscopes framework:

Given an input sequence of n tokens $S = \langle s_1, \ldots, s_n \rangle$ and a model $\mathcal{M}$ with L layers, h_i^ℓ denotes the hidden representation obtained at layer $\ell \in \{1, \ldots, L\}$ and position $i \in \{1, \ldots, n\}$, when running $\mathcal{M}$ on S. To inspect h_i^ℓ, we consider a separate inference pass of a
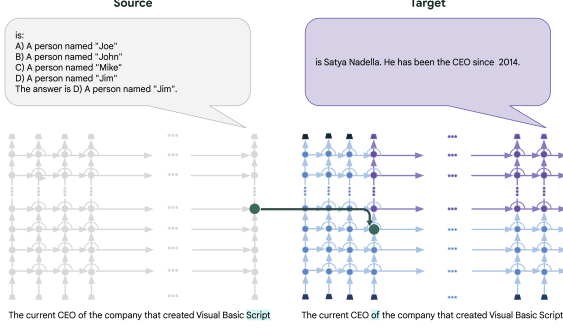
Figure 2: An example of the original Patchscopes framework from Ghandeharioun et al. [2024]

Figure 3: An example of our multi-patching framework

model $\mathcal{M}^*$ with L^* layers on a target sequence $T = \langle t_1, \ldots, t_m \rangle$ of m tokens. Specifically, we choose a hidden representation $h_{i*}^{\ell*}$ at layer $\ell^* \in \{1, \ldots, L^*\}$ and position $i^* \in \{1, \ldots, m\}$ in the execution of $\mathcal{M}^*$ on T. Moreover, we define a mapping function $f(h; \theta) : \mathbb{R}^d \mapsto \mathbb{R}^{d^*}$ parameterized by θ that operates on hidden representations of $\mathcal{M}$, where d and d^* denote the hidden dimension of representations in $\mathcal{M}$ and $\mathcal{M}^*$, respectively. This function can be the identity function, a linear or affine function learned on task-specific pairs of representations, or even more complex functions that incorporate other sources of data. The patching operation refers to dynamically replacing the representation $h_{i*}^{\ell*}$ during the inference of $\mathcal{M}^*$ on T with $f(h_i^\ell)$. Namely, by applying $h_{i*}^{\ell*} \leftarrow f(h_i^\ell)$, we intervene in the generation process and modify the computation after layer ℓ^*.

Overall, a *Patchscopes* intervention applied to a representation determined by $(S, i, \mathcal{M}, \ell)$ is defined by a quintuplet $(T, i^*, f, \mathcal{M}^*, \ell^*)$ of a target prompt T, a target position i^* in this prompt, a mapping function f, a target model $\mathcal{M}^*$, and a target layer ℓ^* of this model. It is possible that $\mathcal{M}$ and $\mathcal{M}^*$ are the same model, S and T are the same prompt, and f is the identity function I (i.e., $I(h) = h$). Next, we show how this formulation covers prior interpretability methods and further extends them with new capabilities.

For instance, Figure 2 illustrates a basic Patchscopes setup for decoding the information encoded in the representation of "CEO" in the source prompt (left). We patch a target prompt (right), which consists of few-shot demonstrations
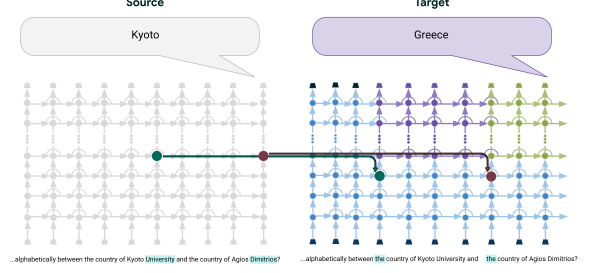
of token repetitions, thereby encouraging the decoding of the token identity from the given hidden representation. As detailed in Ghandeharioun et al. [2024], the levels of the framework are as follows:

1. Execute the forward computation on the source prompt within the source model.

2. Optionally transform the representation from the source layer.

3. Execute the forward computation on the target prompt up to the target layer within the target model.

4. Patch the target representation of "?" at the target layer by replacing it with the transformed representation (from Step 2), and continue the forward computation from that layer onward.

2.2 Modified Framework

In this research, we extend the Patchscopes framework. We propose two extensions to the Patchscopes framework:

2.2.1 Parallel Multi-Patching

We introduce multiple patching (see 3 for a visual representation). We can use the same formulation as in the original framework but change the definition of the positions i, i^*. We each position becomes a pair of positions $(i_1, i_2), (i_1^*, i_2^*)$ such that $i_1, i_2, i_1^*, i_2^* \in \{1, \ldots, n\}$. Now the modified Patchscopes intervention is performing to Patchscopes interventions simultaneously. That is dynamically replacing the representation $h_{i_1*}^{\ell*}, h_{i_1*}^{\ell*}$ during the inference of $\mathcal{M}^*$ on T with $f(h_{i_1}^\ell), f(h_{i_2}^\ell)$ respectively. Namely, by applying $h_{i_1^*}^{\ell*} \leftarrow f(h_{i_1}^\ell), h_{i_2^*}^{\ell*} \leftarrow f(h_{i_2}^\ell)$, we intervene in the generation process and modify the computation after layer ℓ^*. We can further generalize this to k simultaneous patches in this project we only use $k = 2$.

Table 1: Answer Accuracy of Different Techniques

Question Type	Baseline	CoT	1 Patch	2 Patches
Factual multi-hop	3.0%	4.1%	16.0%	**21.0%**

2.2.2 Sequential Patching

In this setting, we do the Patching intervention multiple times sequentially. The process involves applying a series of patching interventions, each modifying the representation at a specific layer and position before proceeding to the next. This approach allows for a more detailed and progressive modification of the representation, accommodating complex transformations that require multiple steps. Here is the step-by-step process:

1. Execute the forward computation on the source prompt within the source model.

2. Optionally transform the representation from the source layer.

3. Execute the forward computation on the target prompt up to the target layer within the target model.

4. Apply the first patching intervention at the target layer by replacing the target representation with the transformed representation from the source layer.

5. Continue the forward computation from the first patched layer onward.

6. Apply subsequent patching interventions at subsequent layers as needed, each time replacing the target representation with the transformed representation from the previous step.

This sequential patching approach allows for the incremental transformation of representations, providing a nuanced method to interpret and modify the internal computations of large language models. By applying multiple interventions in sequence, we can better understand how different layers and positions contribute to the overall representation and output of the model.

3 Experiments

3.1 Data Generation

We utilized several datasets from the original Patchscopes paper [Ghandeharioun et al., 2024]. From these datasets, we extracted subquestions and their corresponding answers and combined each pair of subquestions into a single, comparison-based question.

For example, given the following subquestions and their answers:

1. What is the capital city of England? Answer: *London*

2. What is the capital city of Germany? Answer: *Berlin*

The resulting comparison-based question was: What is first alphabetically between the capital of England and the capital of Germany? Answer: *Berlin*

To ensure the language model responded appropriately, we employed a few-shot learning approach by preceding each subquestion with two additional questions and answers. For instance: *What is the capital city of Israel? Answer: Jerusalem. What is the capital city of France? Answer: Paris. What is the capital city of England? Answer:* We applied a similar method to the composed comparison-based questions.

3.2 Data Filtering

In our constructed dataset, we initially verified that the model can answer the subquestions of the comparison question. For the experiments, we exclusively utilized comparison-based questions composed of subquestions that the model answered correctly. After the filtering process, we used 531 of the remaining examples. We picked only 531 examples because for the current patching Framework we need to test all 40×40 combinations for source layer and target layer. So overall there are 850k generations.

3.3 Model

The Vicunna 13B model was used for all experiments.

3.4 Baseline Experiment

Following the data filtering process on our constructed dataset, we evaluated the model's performance on the comparison-based questions. The model answered correctly on 3% of these questions.
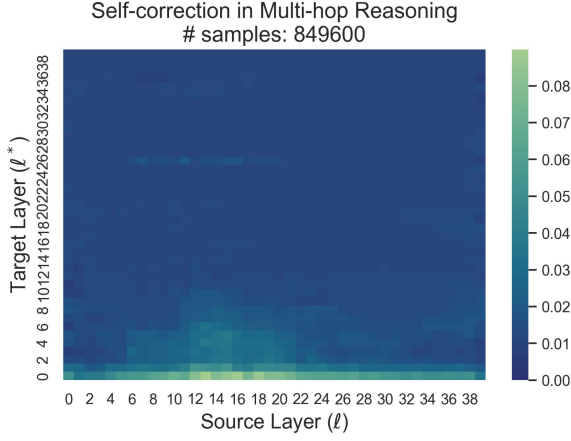
Figure 4: A heat map representing the accuracy percentage achieved on the dataset using the *original* Patchscopes framework. The x-axis represents the source layer and the y-axis represents the target layer of the patching.
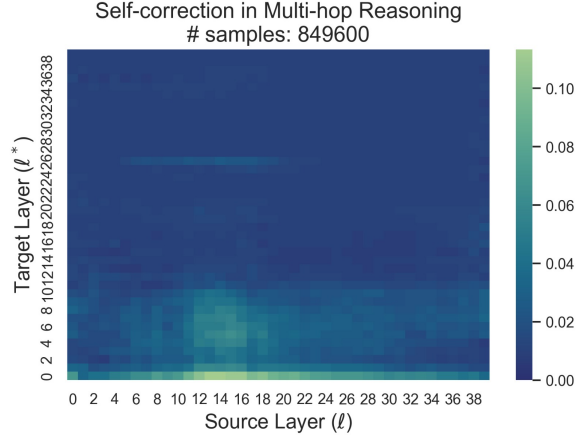

Figure 5: A heat map representing the accuracy percentage achieved on the dataset using the *modified* Patchscopes framework. The x-axis represents the source layer and the y-axis represents the target layer of the patching.

3.5 Chain-of-Thought reasoning

We followed the prompting method laid out in Wei et al. [2022] to test whether chain-of-thought reasoning could help the model solve the complicated multi-hop question by stating the three individual steps required for the solution: answer subquestion A, answer subquestion B, and answer the combined question. We only saw marginal results, improving the baseline from 3% to 4.1%.

3.6 Original Patchscopes Framework on New Structured Data

This experiment involved applying the original Patchscopes framework, as introduced in Ghandeharioun et al. [2024], to our newly constructed dataset. For each comparison-based question, we attempted to "patch" the encoding state from the token location at the end of the first subquestion to the token location at the end of the comparison question. For example, in the question "What is first alphabetically between the capital of England and the capital of Germany?", we patched the token corresponding to "England" to the token corresponding to "alphabetically".

We performed this patching between each pair of layers in the model. Using this framework, the model correctly answered 16% of the questions. Figure 4 illustrates a heat map representing the accuracy of applying the patching between each pair of layers in the model.

3.7 Parallel Multi-Patching

This experiment involved applying the modified Patchscopes framework, Parallel Multi-Patching, as explained in the method section, to our constructed dataset. For each comparison-based question, we performed two patches:

1. We attempted to "patch" the encoding state from the token location at the end of the first subquestion to the token location at the end of the comparison question.

2. We attempted to "patch" the encoding state from the token location at the end of the second subquestion to the token location at the end of the comparison question.

For example, in the question "What is first alphabetically between the capital of England and the capital of Germany?", we patched the token corresponding to "England" to the token corresponding to "alphabetically", and the token corresponding to "Germany" to the token corresponding to "alphabetically".

We performed this patching between each pair of layers in the model. Using this framework, the model correctly answered 21% of the questions. Figure 5 illustrates a heat map representing the accuracy of applying the patching between each pair of layers in the model.

3.8 Sequential Patching + Multi-Patching

This experiment involved applying the last experiment with two generation. Meaning we use the

same configuration for the patching source and target indexes but we do two generations sequentially. Due to computational limitations we use the same source and target layer tuples for both generations. So we go over all 40×40 combinations, but each time they are the same for both generations. Our results showed that there was no improvement from applying only the multi-patching. We still reach 21% accuracy.

4 Discussion

Applying the original Patchscopes framework to the comparison-based questions resulted in a 13% improvement in the model's accuracy. The modified Patchscopes framework, which is better suited to the comparison structure, further improved accuracy by an additional 5%. These results indicate that the Patchscopes framework has a positive impact on answering comparison-based questions. Moreover, it highlights that modifying the framework to better fit the question structure can lead to even greater improvements.

4.1 Next Steps

1. Explore both the original and modified Patchscopes frameworks on larger datasets.

2. Investigate the application of the Patchscopes framework on different structures of questions, as outlined in Table 1.

3. Develop a classifier to determine the source and target layers for patching based on a given question, enabling the framework to operate automatically. We believe that this automated process can improve the results of the modified framework significantly. As for now, we use the same source and target layers for all patches.

5 Acknowledgements

We would like to express our gratitude to Mor Geva for her guidance throughout this research.

References

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. Patchscopes: A unifying framework for inspecting hidden representations of language models. 2024. URL https://arxiv.org/pdf/2401.06102v1.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. 2022. URL https://arxiv.org/pdf/2108.00573.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2022.