Frameworks for Mechanistic Interpretability

Shaowei Zhang

Uni Heidelberg

23.1.2024

Outline

Overview

Main Purpose: Understand Representations of LLMs

Two Methods:

- InversionView[1]: Decode Information from Activations
- Patchscopes[2]: Explain Representations in Natural Language

Review

- Activation Patching

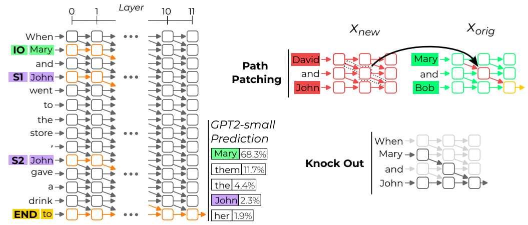

Figure 1: A structure of activation patching in [3]

Review

- Anto Encoder [4]
- Probing Classifiers[5]: Predict some linguistic property from a model's representations.
- IOI [3][6]
  Example: "When Mary and John went to the store", "John gave a bottle of milk to ?".

InversionView

Main Question: What information is encoded by an activation in a neural network?

InversionView

Main Question: What information is encoded by an activation in a neural network?
InversionView aims to find those inputs that **give rise to the same activation**, and examine what's common among them to infer what information it encodes.

InversionView

Main Question: What information is encoded by an activation in a neural network?

InversionView aims to find those inputs that **give rise to the same activation**, and examine what's common among them to infer what information it encodes.

Larger changes make it easier for downstream components to read out information than very small changes.

ϵ-preimage

Given a space $\mathcal{X}$ of valid inputs, a query input $x^q \in \mathcal{X}$, a function f that represents the activation of interest as a function of the input, and a query activation $z^q = f(x^q)$, define the ϵ-*preimage*:

$$B_{z^q,f,\epsilon} = \{x \in \mathcal{X} : D(f(x), z^q) \leq \epsilon\}, \tag{1}$$

where $\epsilon > 0$ is a threshold and $D(\cdot, \cdot)$ is a distance metric.

ϵ-preimage

When x^q is a sequence, they study the vector z^q corresponding to a specific position t in this sequence, i.e. $z^q = f(x^q)_t$ where $f(x^q)_t$ represents taking the activation from the site of interest (abstracted by f) at position t in input sequence x^q.

ϵ-preimage:

$B_{z^q,f,\epsilon} = \{x : x \in \mathcal{X}, \exists t \in [1, |x|] : D(f(x)_t, z^q) \leq \epsilon\}$.

Conditional Decoder Model

Autoencoder Conditional Decoder Model: A LLM trained to **recover** x^q **from** z^q.

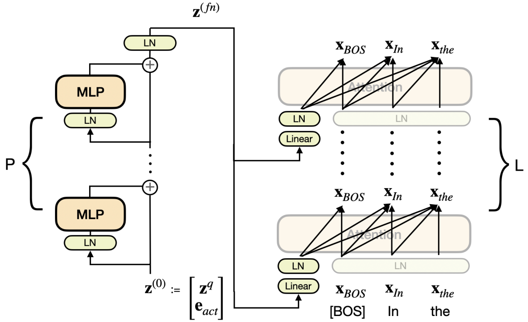

Figure 2: The decoder model architecture used in InversionView.

Conditional Decoder Model

The query activation $z^q \in \mathbb{R}^d$ is first concatenated with a trainable activation site embedding $e_{act} \in \mathbb{R}^{d_{site}}$, producing the intermediate representation $z^{(0)} = [z^q; e_{act}]$. d_{site} is the number of possible activation sites in the training set.

Experimental Setup

The residual stream: $x^{i,\{\text{pre},\text{mid},\text{post}\}} \in \mathbb{R}^{N\times d}$, where i is the layer (an attention (sub)layer + an MLP (sub)layer) index, and pre, mid, post stand for the residual stream before the attention layer, between attention and MLP layer, and after the MLP layer.
$x^{i,\text{post}} = x^{i+1,\text{pre}}$.
$x_t^{i,\text{mid}} \in \mathbb{R}^d$ is the activation at token position t.
The attention layer output decomposes into outputs of individual heads $h^{i,j}(\cdot)$, i.e., $x^{i,\text{mid}} = x^{i,\text{pre}} + \sum_j h^{i,j}(\text{LN}(x^{i,\text{pre}}))$.
$a^{i,j}$ is attention head's output, i.e., $a^{i,j} = h^{i,j}(\text{LN}(x^{i,\text{pre}}))$.

InversionView on Character Counting Task

Task Example: "vvzccvczvvvzvcvc|v:8"
To predict the frequency of the target character (here, "v") before the separator "|".
A transformer with 2 layers and 1 head is trained for this task.

$D(z, z^q) = \frac{\|z - z^q\|_2}{\|z^q\|_2}$ (i.e., normalized euclidean distance)
$\epsilon = 0.1$

InversionView on Character Counting Task

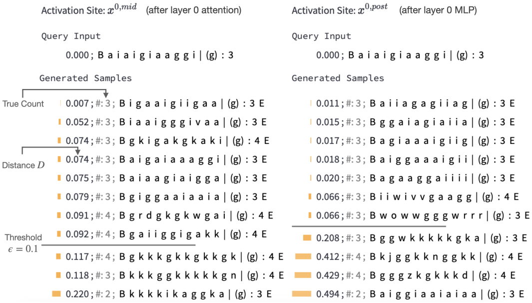

Figure 3: InversionView on Character Counting Task.

InversionView on Character Counting Task

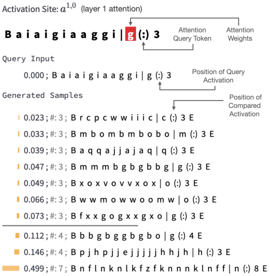

Figure 4: InversionView on Character Counting Task.

InversionView on IOI Task

Task Example: "When Mary and John went to the store, John gave a drink to"– "Marry"
A GPT-2 small is trained for this task.

$D(z, z^q) = 1 - \frac{z \cdot z^q}{||z|| \cdot ||z^q||}$ (i.e., cosine distance), and $\epsilon = 0.1$.

InversionView on IOI Task

<|endoftext|>After Erin and Justin went to the house, Erin gave a ring(to) Justin

Query Input

0.000 ; <|endoftext|>After Erin and Justin went to the house, Erin gave a ring(to) Justin

Generated Samples

0.024 ; <|endoftext|>The station Sara and Justin went to had a kiss. Sara gave it(to) Justin[EOS]

0.024 ; <|endoftext|>When Paul and Justin got a kiss at the school, Paul decided to give it(to) Justin[EOS]

0.025 ; <|endoftext|>Then, Alicia and Justin had a long argument. Afterwards Alicia said(to) Justin[EOS]

0.034 ; <|endoftext|>Then, Justin and Erin went to the garden. Erin gave a basketball(to) Justin[EOS]

0.037 ; <|endoftext|>After the lunch in the afternoon, Justin and Kristen went to the station. Kristen gave a kiss(to) Justin[EOS]

0.039 ; <|endoftext|>After taking a long break Kimberly and Justin went to the house, Kimberly gave a bone(to) Justin[EOS]

0.043 ; <|endoftext|>While spending time together Justin and Alicia were working at the garden, Alicia gave a kiss(to) Justin[EOS]

0.056 ; <|endoftext|>Then, Justin and Kristen went to the school. Kristen gave a bone(to) Justin[EOS]

0.506 ; <|endoftext|>Friends separated at birth Kristen and Justin found a snack at the garden. Justin gave it(to) Kristen[EOS]

0.598 ; <|endoftext|>While spending time together Michelle and Joshua were commuting to the restaurant, Alexander gave a ring(to) Michelle[EOS]

Figure 5: InversionView applied to Name Mover Head 9.9 at “to”

InversionView on 3-Digit Addition

Task Example: "B362+405=767E" and "B824+692=1516E"
F1, F2, F3 denote the three digits of the first operand.
S1, S2, S3 for the digits of the second operand.
A1, A2, A3, A4 (if it exists) for the three or four digits of the answer.
C2, C3 for the carry from tens place and ones place.
A decoder-only transformer (2 layers, 4 attention heads, dimension 32) is trained for this task.
$D(z, z^q) = \frac{\|z - z^q\|_2}{\|z^q\|_2}$ (i.e., normalized euclidean distance) and $\epsilon = 0.1$.

InversionView on 3-Digit Addition

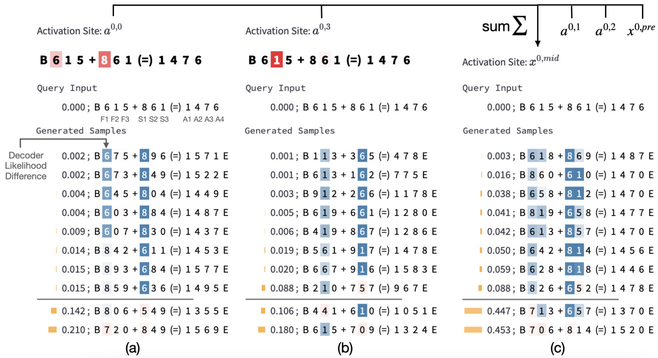

Figure 6: InversionView applied to 3-digit addition

Summary on InversionView

- InversionView: A tool for decode activations.
- Detailed experiments about what can you do with InversionView.
- InversionView can scale from 2-layer transformer to GPT-2 XL.

Limitations:

- Relying on a black-box decoder.
- Not clear whether it can work when models are larger.
- Decoding some irrelevant information.

Patchscopes

Key idea: To leverage the advanced capabilities of LLMs to generate human-like text for “translating” the information encoded in their own hidden representations.

Algorithm of Patchscopes

Given an input sequence of n tokens $S = \langle s_1, ..., s_n \rangle$ and a model $\mathcal{M}$ with L layers, $\boldsymbol{h}_i^\ell$ 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 $\boldsymbol{h}_i^\ell$, they consider a separate inference pass of a 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 $\bar{\boldsymbol{h}}_{i^*}^{\ell^*}$ 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 $f(\boldsymbol{h}; \boldsymbol{\theta}) : \mathbb{R}^d \mapsto \mathbb{R}^{d^*}$ parameterized by $\boldsymbol{\theta}$ 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.

The *patching* operation refers to dynamically replacing the representation $\bar{\boldsymbol{h}}_{i^*}^{\ell^*}$ during the inference of $\mathcal{M}^*$ on T with $f(\boldsymbol{h}_i^\ell)$.

Algorithm of Patchscopes

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 $\mathbb{I}$ (i.e., $\mathbb{I}(\boldsymbol{h}) = \boldsymbol{h}$).

Algorithm of Patchscopes

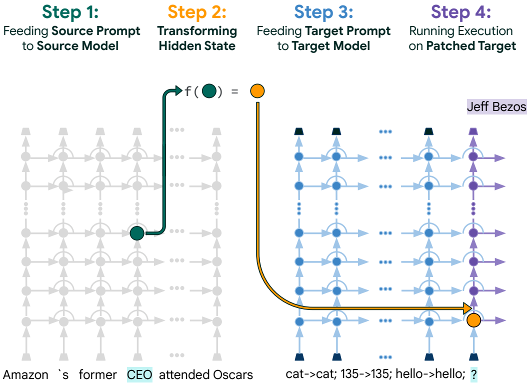

Figure 7: Procedure of a Patchscope

Patchscopes as a Framework

Table 1. Many prior inspection methods with various objectives can be viewed as Patchscopes. The rows highlighted in green show Patchscope configurations that overcome several limitations of prior methods through more expressive inspection that is training-data free and is more robust across layers.

Inspection Objective		Expressive	Training Data Free	Robust Across Layers
Inspecting Output Distribution	Few-Shot Token Identity `Patchscope` (§4.1)	✔✔	✔	✔✔
	Logit Lens (nostalgebraist, 2020), Embedding Space Analysis (Dar et al., 2023)	✔	✔	✖
	Tuned Lens (Belrose et al., 2023)	✔	For learning mappings	✔
	Future Lens (Pal et al., 2023)	✔	For learning mappings	✔✔
Feature Extraction	Zero-Shot Feat. Ext. `Patchscope` (§4.2)	✔✔	✔	✔✔
	LRE Attribute Lens (Hernandez et al., 2023b)	✔	For linear relation approx.	✔✔
	Probing (e.g., Belinkov & Glass, 2019; Belinkov, 2022; Alain & Bengio, 2017; Wang et al., 2023)	✖	For training probe	✔
Entity Resolution	Entity Description `Patchscope` (§4.3)	✔✔	✔	✔✔
	X-Model Entity Desc. `Patchscope` (§4.4)	✔✔✔	For learning mappings	✔✔
	Causal Tracing (Meng et al., 2022a)	✖	✔	✔✔
	Attention Knockout (Wang et al., 2022; Conmy et al., 2023; Geva et al., 2023)	✖	✔	✔✔
Inspection Application	Early Exiting, e.g., Linear Shortcuts (Din et al., 2023)	✔	For learning mappings	✔
	Caption Generation, e.g., Linear Mapping (Merullo et al., 2022)	✔	For learning mappings	✔

Figure 8: Prior methods can be viewed as Patchscopes.

Patchscopes on Next-Token Prediction Decoding

To estimate the output probability distribution from hidden representations from a model.

Baselines

- Logit Lens: No Change
- Tuned Lens: Learnable Linear Projection
- Patchscope: $\mathcal{M} = \mathcal{M}^*$,$T \neq S$,
  Target Prompt: "$\texttt{tok}_1 \rightarrow \texttt{tok}_1$; $\texttt{tok}_2 \rightarrow \texttt{tok}_2$; ... ; $\texttt{tok}_k$"

Patchscopes on Next-Token Prediction Decoding

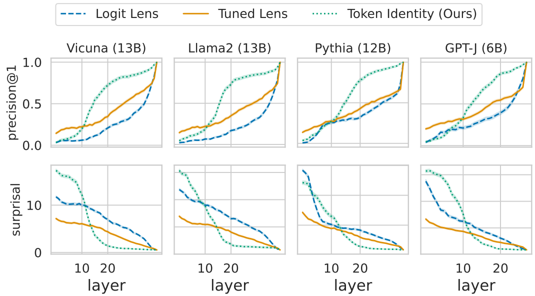

Figure 9: Precision@1 (↑ is better) and Surprisal (↓ is better) of next-token prediction estimation in multiple models

Patchscopes on Extraction of Specific Attributes

A triplets: (σ, ρ, ω) of a subject (e.g., "`United States`"), a relation (e.g., "`largest city of`"), and an object (e.g., "`New York City`").

How to extract the object ω from the last token representation of the subject σ in an arbitrary input context.

Patchscopes on Extraction of Specific Attributes

Methods:
1. Logistic Regression Probe
2. Zero-shot Feature Extraction Patchscope: $\mathcal{M}^* = \mathcal{M}$. Target Prompt: "`The largest city in x`" with "`x`" as a placeholder for the subject.

Evaluation: For a given sample, the `Patchscope` is considered correct if $\exists\, \ell^* \in [1, \ldots, L^*]$ where **the generated text up to 20 tokens includes** ω. For the probe, a prediction is correct if the highest probability is assigned to ω.

Patchscopes on Extraction of Specific Attributes

Table 2. Feature extraction accuracy (mean±std). Comparing zero-shot feature extraction `Patchscope` to a logistic regression probe shows that despite using *no training data*, it has a significantly higher accuracy than baseline in 6 out of 12 tasks. We use pairwise t-test with Bonferroni correction for comparing the two methods. ** and * indicate $p < 1e{-5}$ and $p < 1e{-4}$, respectively.

	Task	Probe	Patchscope
Commonsense	Fruit inside color	37.4 ± 6.6	38.0 ± 18.7
	Fruit outside color	35.5 ± 3.1	$\mathbf{71.0 \pm 13.3}^{**}$
	Object superclass	$\mathbf{65.6 \pm 10.5}^{*}$	54.8 ± 11.3
	Substance phase	73.8 ± 3.7	$\mathbf{91.9 \pm 1.7}^{**}$
	Task done by tool	10.1 ± 3.2	$\mathbf{48.1 \pm 13.2}^{**}$
Factual	Company CEO	5.0 ± 2.6	$\mathbf{47.8 \pm 13.9}^{**}$
	Country currency	17.7 ± 2.2	$\mathbf{51.0 \pm 8.9}^{**}$
	Food from country	5.1 ± 3.7	$\mathbf{63.8 \pm 11.3}^{**}$
	Plays pos. in sport	75.9 ± 9.1	72.2 ± 7.2
	Plays pro sport	53.8 ± 10.3	46.3 ± 14.2
	Product by co.	58.9 ± 7.2	63.2 ± 10.7
	Star constellation	17.5 ± 5.3	18.4 ± 5.1

Figure 10: Feature extraction accuracy (mean±std).

Analyzing Entity Resolution in Early Layers

Task: Describe some entities. Example: "Wales : Country in the United Kingdom"

Few-shot target prompt template for Patchscopes:

"$\texttt{subject}_1\texttt{: description}_1\texttt{, ..., subject}_k\texttt{: description}_k\texttt{,}$ $\texttt{x}$", while patching the last position corresponding to $\texttt{x}$.

Analyzing Entity Resolution in Early Layers

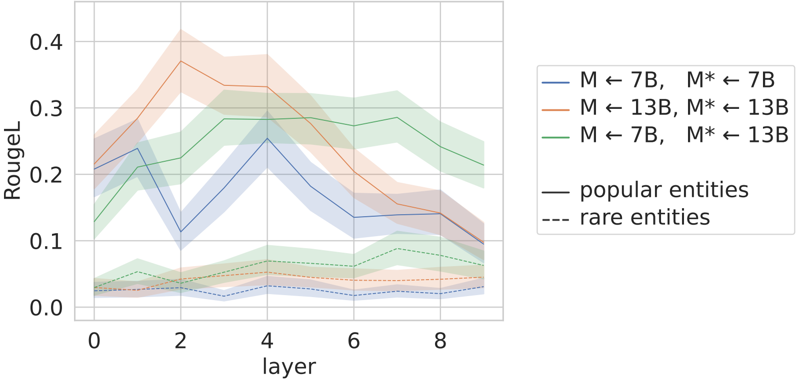

Figure 11: RougeL scores of the generated descriptions against descriptions from Wikipedia, using Vicuna models.

Expressiveness from Cross-Model Patching

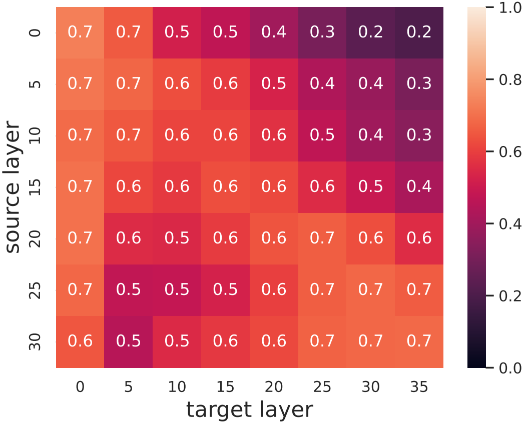

Figure 12: Precision@1 scores (↑ is better) on next-token prediction estimation in Vicuna with cross-model `Patchscopes`

Application: Correcting Multi-Hop Errors

Task Description: For example, Let $\tau_1 \leftarrow$ (“`Visual Basic`”, “`product of`”, “`Microsoft`”) and $\tau_2 \leftarrow$ (“`Microsoft`”, “`company CEO`”, “`Satya Nadella`”). An example verbalization of these tuples is $\pi_1 \leftarrow$ “`the company that created Visual Basic`”, $\pi_2 \leftarrow$ “`The current CEO of`”, leading to the multi-hop query $[\pi_2][\pi_1] =$ “`The current CEO of the company that created Visual Basic`”.

Application: Correcting Multi-Hop Errors

Methods:

- Vanilla Baseline: set $S \leftarrow [\pi_1][\pi_2]$, we let the model autoregressively generate up to 20 tokens and check whether ω_2 appears in the generation.
- Chain-of-Thought Baseline: Here, the setup and evaluation is similar to the vanilla baseline, except that we prepend "Let's think step by step." to S.
- Patchscope: $T \leftarrow S, \mathcal{M}^* \leftarrow \mathcal{M}$,
  Source: The current CEO of the company that created Visual Basic **Script**.
  Target: The current CEO **of** the company that created Visual Basic Script.

Application: Correcting Multi-Hop Errors

Table 1: Comparison of Methods for Multi-Hop Reasoning

Method	Accuracy
Vanilla Baseline	19.57%
CoT Baseline	35.71%
Patchscope	50%

Summary on Patchscopes

- Patchscopes: a simple and effective framework that leverages the ability of LLMs to generate humanlike text for decoding information from intermediate LLM representations.
- A real framework that can contain familiar methods.
- It shows the changing information of representations by layers.
- It can really scale to larger models.

Summary

- InversionView: A method which can decode activations.
- Patchscopes: A framework which help LLMs to generate leveraging LLM representations.

References I

[1] Xinting Huang et al. *InversionView: A General-Purpose Method for Reading Information from Neural Activations*. 2024. arXiv: 2405.17653 [cs.LG]. URL: https://arxiv.org/abs/2405.17653.

[2] Asma Ghandeharioun et al. *Patchscopes: A Unifying Framework for Inspecting Hidden Representations of Language Models*. 2024. arXiv: 2401.06102 [cs.CL]. URL: https://arxiv.org/abs/2401.06102.

[3] Kevin Wang et al. *Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 small*. 2022. arXiv: 2211.00593 [cs.LG]. URL: https://arxiv.org/abs/2211.00593.

References II

[4] Trenton Bricken et al. *Towards Monosemanticity: Decomposing Language Models With Dictionary Learning*. Tech. rep. Transformer Circuits Thread, 2023. URL: https://transformercircuits.pub/2023/monosemantic-features/index.html.

[5] Yonatan Belinkov. *Probing Classifiers: Promises, Shortcomings, and Advances*. 2021. arXiv: 2102.12452 [cs.CL]. URL: https://arxiv.org/abs/2102.12452.

[6] Arthur Conmy et al. *Towards Automated Circuit Discovery for Mechanistic Interpretability*. 2023. arXiv: 2304.14997 [cs.LG]. URL: https://arxiv.org/abs/2304.14997.

Q&A I

- Is Patchscopes decoding the latent space of LLM? And how is this "decoding" serving as interpretation? And can this framework be extended from layer level to more complicated structures?
- Patchscopes: Where do the variations in performance in the feature extraction task come from, when you look at Table 2, the Object superclass vs The Substance phase for example?
- Using the model's own ability to decode hidden representations is very creative. The target prompt design is a strong point of this framework. But for different tasks, prompts need to be adjusted manually. This makes it less convenient to use. Is it possible to find a way to create efficient prompts automatically?

Q&A II

- The framework allows using different mapping functions like identity mapping or linear mapping. Would more complex nonlinear functions work better?
- What are the limitations of using a conditional decoder model to approximate the -preimage, and how might these limitations impact the validity of the interpretations?
- The conditional decoder model considers both the geometry of activations and adds diversity by using temperature and noise. This design makes the samples accurate and covers a wider preimage space. However, I am worried that training the decoder could be complicated.

Thanks

Thank you for your Attention!