

Reprogramming AI Models Hackathon Submission

Big Brain Large Language Model (BBLLM)

Joey Skaf
joey.skaf@outlook.fr

Mickaël Boillaud
mickael.boillaud@gmail.com

Thaïs Distinguin
thaisdstng@gmail.com

With
Goodfire AI and Apart Research

Abstract

Feature interpretability represents a significant step toward unraveling AI's black box. However, the relationships between latent features remain challenging to conceptualize and represent intuitively. To tackle this, we developed an approach to uncover the structure between latent features by constructing an interactive graph that connects features co-activated by a given prompt. We deployed this graph-based representation on a web application, for Llama-3-70B and Llama-3-8B, to encourage a user-friendly comparison and exploration of feature clusters created by a sample prompt sentence, for each word. Key results include the identification of clusters of co-activated features, which provide a window into latent feature organisation and their potential roles in decision-making processes.

Keywords: AI Observability, Mechanistic Interpretability

1 Introduction

Understanding the inner workings of AI systems is crucial for ensuring value alignment and safety. This project focuses on AI observability through a web app designed to provide intuitive, brain-like visualizations of large language models (LLMs). By emphasizing the central yet elusive concept of features in neural networks, this app highlights their correlations and relevance, making mechanistic interpretability more accessible.

These insights could not only be used for our understanding of the model behavior but also support better labeling and alignment practices, offering a tool for navigating the pre-paradigmatic state of interpretability.

2 Overview

2.1 General idea

Our project stems from the following question: how could we observe the entirety of accumulated LLM knowledge such as a brain, including the strong connections and clusters of notions formed through repeated activations?

With access to labeled feature activation information for selected tokens, this inquiry inspired the development of an algorithm capable of generating a co-occurrence graph. This graph highlights the strong connections and clusters of concepts formed through numerous occurrences of feature activations, shedding light on the underlying structure of latent knowledge within the model.

This approach is centered on human-interpretability but could also provide guidance for a non-human-centric perspective that defines features as independent model components, whether aligned with human concepts or not, revealing connections between latent spaces within AI models.

2.2 Graph structure and generation

The dataset selection was driven by the objective of capturing diverse tonalities and themes in conversations, tailored to stimulate the type of feature activations labeled by Goodfire’s team. This deliberate choice ensured the dataset would effectively highlight the nuanced connections and activations within the models. We chose sharegpt-english dataset [3] that is a database of conversations between a user and an AI assistant. We focused on a token-level to retrieve the features due to their context sensitivity, decided through some experiments conducted on feature activations of the token "A" depending of its context (cf. Figure 2).

To construct a meaningful graph representation, a thresholding process was applied to filter out edges with weights below a certain value, while also selecting the most connected subgraph. This approach effectively removed poorly-connected and isolated clusters, ensuring the graph focused on significant and interpretable relationships. As a result, a cohesive structure was created that emphasized the most relevant latent connections.

We then applied the Louvain method [1] to identify distinct clusters, enabling the observation of coherent communities and raising new questions about the relationships and structures within the latent space. This community analysis offered deeper insights into how features interact and group together, providing a rich foundation for further exploration.

2.3 Application Interface

To make this research accessible and intuitive, we designed a web application with an educational focus. The interface features dual windows to facilitate side-by-side model comparison, emphasizing latent space connections and engaging users beyond the scientific community. This user-friendly design underscores the project’s broader goal of making feature interpretability accessible to all.

3 Methodology and Approach

Our project integrates various tools and libraries to process data, build graph representations, and enable visualization:

- **Goodfire:** Used to retrieve features and build graphs.
- **NetworkX:** Utilized for graph data manipulation and visualization.
- **React, Sigma.js, Graphology:** Implemented for front-end visualization and interactivity.

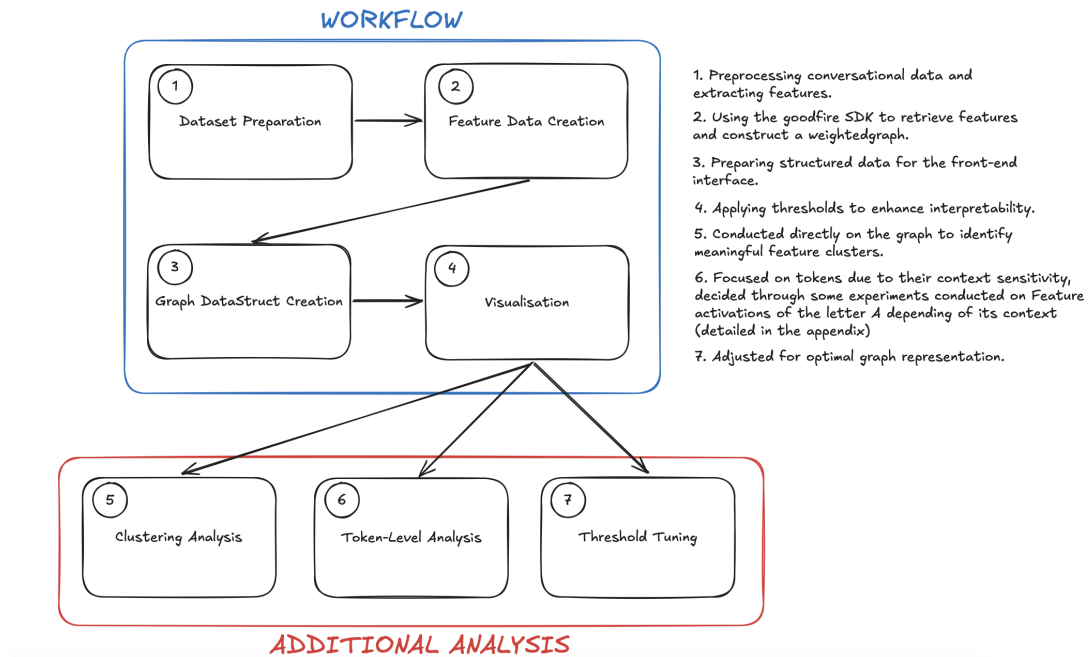


Figure 1: Schema of the project workflow. For the clustering analysis, refer to videos contained in the following folder : https://drive.google.com/drive/folders/185qflptYZqZZ8e0x9HQAUWst90Ip9d-N?usp=share_link.

3.1 Workflow

3.2 Code sample

The following snippet illustrates the main steps for extracting features, constructing the graph, and preparing data for visualization:

```

context = client.features.inspect(conversation_data, model=variant)

tokens_features_list = []
for token in tqdm(context.tokens):
    if not ("<|" in str(token) and "|>" in str(token)):
        tokens_features_list.append(
            [token._token, [
                {"feature_label": act.feature.label,
                 "feature_index_sae": act.feature.index_in_sae,
                 "activation": act.activation}
                for act in token.inspect()._acts if act.activation > THRESHOLD]]
        )
    node_feature_ids = [act.feature.index_in_sae
                        for act in token.inspect()._acts
                        if act.activation > THRESHOLD]
    for u, v in combinations(node_feature_ids, 2):
        if graph.has_edge(u, v):
            graph[u][v]["weight"] += 1
        else:
            graph.add_edge(u, v, weight=1)
    conversational[i]["token_features"] = tokens_features_list
  
```

For more details and access to the project code:

3.3 Results & Analysis - Visualisation tool review

At its core, the visualisation model we developed is full of inherent biases. As the study focuses on answering a specific question, with a human-centered representation, we influenced by a biased dataset and a double-thresholding approach—both at the graph creation stage and during visualization. Additionally, the biases induced from prior labeling are accumulating in our model, and are also to take into account in our conclusions.

The integration of clustering into the visualization interface enhances interpretability but remains highly sensitive to the dataset used for construction. Videos of the results illustrate this point: while most clusters appear meaningful, some could be mitigated by a better dataset quality. The weights also influence the Louvain method applied for clustering.

Comparison between models reveals significant challenges due to the varying number of features extracted for each. This variation, combined with small discrepancies in labeling practices, complicates meaningful comparisons. Furthermore, issues stemming from the dataset used to construct the graphs are apparent; for instance, the dataset for the 70B model included 300 additional conversations, disproportionately emphasizing features like “repetitive or corrupted text patterns” and “template placeholders or numbered sequences in generated content.” Lastly, bias is also introduced by the labeling of features and those identified by SAE methods, which impacts the interpretation and evaluation of results.

4 Discussion and Conclusion

4.1 Implications for AI Interpretability

Our hackathon project has demonstrated several key findings and potential avenues for future work. Indeed, we have shown that the model is capable of forming links between words and features, allowing us to observe feature neighborhoods and their connections. For instance, for the word “treasure,” as shown in Figure 3. Our tool also provides an extensive understanding of the relationships between features through clustering analysis.

The comparative analysis between the two models further highlighted the sensitivity of graph representation to the dataset used for construction. This underscores the need for datasets that are diverse and representative of what the model typically encounters. For instance, the 70B model may appear more accurate because its dataset includes generic features frequently observed by the LLM, such as “Repetitive or corrupted text” providing a perspective aligned with the model’s “point of view.”

The clustering part should be reconsidered through a better construction of the graph, as it is highly sensitive to the biases discussed earlier. We could explore a more effective mathematical representation for the graph, such as including the activation values (which we store in a database to avoid re-running the entire process) in the calculation of the edge weights. This visualization tool could also enable a better selection of the layer where SAEs are trained by comparing the quality of features induced by multiple layers using our graph representation.

4.2 Future Work

In terms of potential expansions, leveraging API functionalities for dynamic prompt-based requests could further refine insights. Additionally, using a larger dataset to better capture feature activation patterns would be invaluable for improving labeling practices. Future work could also explore advanced techniques such as optimal flow or optimal transport on graphs, potentially yielding more nuanced and effective representations. On the API level, one could continue to add the functionalities developed by the Ipsigma project at the Media Lab to enhance the

Web App front. These directions highlight the promise of further developing this framework to deepen our understanding of model behavior and its practical applications.

[3] [4] [2] [1]

References

- [1] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, October 2008.
- [2] Giuseppe Casalicchio, Christoph Molnar, and Bernd Bischl. *Visualizing the Feature Importance for Black Box Models*, page 655–670. Springer International Publishing, 2019.
- [3] theblackcat102. sharegpt-english, 2023.
- [4] Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency, 2023.

5 Appendix

```
In [10]: context = client.features.inspect(
[
    {
        "role": "user",
        "content": "Option A"
    },
],
model=variant,
)
for token in context.tokens:
    print(token._token)
    print(token.inspect())
```

(a) Token "A" with token "Option" prompt.

```
A
FeatureActivations(
  0: (Feature("Multiple choice question first answer option"), 3.515625)
  1: (Feature("References to abstract entities (e.g. tables, classes) in programming contexts"), 1.515625)
  2: (Feature("Portuguese event announcements and invitations"), 0.7734175)
  3: (Feature("Beginning of task or problem description"), 0.76171875)
  4: (Feature("Multilingual summary or conclusion request"), 0.454625)
)
```

(c) Feature activation for the token "A" with context "Option".

```
In [11]: context = client.features.inspect(
[
    {
        "role": "user",
        "content": "Variable A"
    },
],
model=variant,
)
for token in context.tokens:
    print(token._token)
    print(token.inspect())
```

(b) Token "A" with token "Variable" prompt.

```
A
FeatureActivations(
  0: (Feature("Variable 'a' in programming and mathematics"), 2.421875)
  1: (Feature("References to abstract entities (e.g. tables, classes) in programming contexts"), 1.5703125)
  2: (Feature("Ordinal comparisons and rankings, especially involving 'first' or 'best'"), 1.3671875)
  3: (Feature("Common concrete nouns"), 1.234375)
  4: (Feature("Beginning of task or problem description"), 1.1015625)
)
```

(d) Feature activation for the token "A" with context "Variable".

Figure 2: Proof of the token context-sensitivity of feature activation.

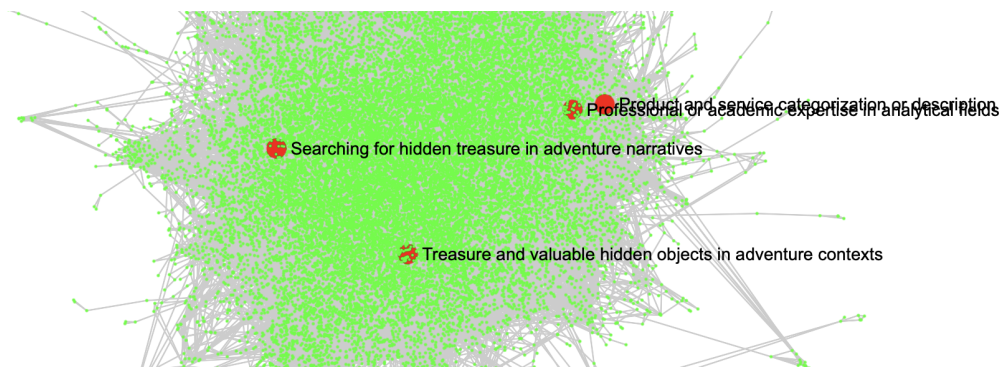


Figure 3: Token context-sensitivity and activation feature response from word "Treasure"