Using Syntax-Based Context Visualizations To Understand Features of Language Models

Eric Xia*, Byron Butaney*, and Gonçalo Paulo**

*Department of Computer Science, Brown University
**EleutherAI

ABSTRACT

By identifying monosemantic features from model weights, Sparse Autoencoders (SAEs) allow for a more complete understanding of how neural language models function. This work introduces two novel methods for unifying SAE feature contexts, one based on syntax trees and one based on linear aggregation. Users found syntactic visualizations promising but confusing; initial survey results demonstrate that our linear aggregation method performed worse than the baseline. The results demonstrate the challenges of (1) employing syntactic methods for feature analysis and (2) facilitating textual comprehension through visualization.

Keywords: Human Computer Interaction, Interpretability, Dependency Parsing, Natural Language Processing, Large Language Models

1 INTRODUCTION

Large language models are becoming increasingly integrated in daily life, but their underlying mechanisms are not fully understood. Recently, sparse autoencoders (SAEs) have emerged as a promising way to extract features from models. [2] These features activate on input contexts in predictable ways, with some exhibiting consistent syntactic patterns. Improved understanding of features through their contexts can facilitate comparisons between features, identify training issues such as over-splitting, and simplify identification of highly syntactic features.

Consequently, this work investigates unified context visualizations for individual SAE features. One critical issue with current feature dashboards is their use of a list of text contexts to characterize a features. Although this may aid in identifying repetition over contexts, characterizing features with textual contexts fail to highlight the linguistic abstractions which features represent.

2 RELATED WORK

Current research in the field of mechanistic interpretability shows that SAEs can successfully train on larger and more capable models, such as Gemma Scope [6] and Claude 3.5 Sonnet [11], providing promising opportunities to advance interpretability. However, achieving feature monosemanticity only serves as a first step in interpretability [2]. Crucial to utilizing features in interpretability is a way to understand the role they serve within language models. For an SAE, features are defined as weighted combinations of neurons from specific layers of the model. Still, many methods have been proposed for extracting features, such as transcoders, [3], crosscoders [10], and Meta SAEs [1]. Feature characterization is critical for any interpretability method which relies on regularly activating patterns in text. Research in the field, as conducted in this work, has long-term implications for interpretability.

There are many specific applications which would benefit from improved feature identification. One common goal within interpretability is to identify universal features across models. These notions of universality require features identified for one model to be compared to others. Through neuron-level comparisons of output contexts, universal activations have been identified across GPT-2 models on punctuation, dates, and medical terms [4]. We build on prior, text-based methods through the creation of merged visualizations, allowing researchers to compare emergent semantic features across models.

Other papers in interpretability that utilize SAE features do so by identifying groups of features that work together. For example, feature comparisons have led to the identification of features where occlusion and over-splitting occur [7]. By unifying text contexts, our visualization aims to provide an increased understanding of structure among features, aiding in identification of occlusion and over-splitting.

Ultimately, understanding both the scope and context of feature activations will be necessary to characterize the performance of interpretability techniques. Current text views fail to identify or aggregate shared contexts and scope. Our work explores methods for characterizing features through visual aggregation.

3 METHODOLOGY

Our context visualizations are implemented with SpaCy, Huggingface, and Plotly. We additionally use Uniform Manifold Approximation and Projection (UMAP) and the Transformer Lens libraries to visualize decoder features through dimensionality reduction. We primarily used data provided by our collaborator, consisting of intermediate-layer feature activations of a JumpReLU SAE, a current state-of-the-art architecture [9]. These activations were on tokens from Google's Gemma-9b model on a eWeb-100m dataset [8].

Two focus areas were identified. The first focus was to aggregate part of speech tags for each feature to improve characterization of the feature space. The second focus was to unify contexts through syntactic methods.

3.1 Feature Navigation

Our tool presents users with a generated UMAP upon initiating the tool. Viewing the feature space in this way allows users to navigate the set of features based on the UMAP clusters, which may be syntactically significant. This provides another layer of information to users that could be used to discriminate between features of potential interest. Users can also highlight a region of the UMAP with their cursor to zoom into the area, allowing for more precise choices between features on the plot. After selecting a feature from the UMAP, users are able to view the contexts upon which the feature activates in multiple views. Having access to this more fine-grained information about the features allows users to determine its relevance and thus further navigate the dataset.

3.2 Feature Views

There are two primary views developed for the study. For each feature, the top 50 activations in their surrounding context sentences are displayed. Our syntactic trees are created using SpaCy's dependency parser [1] [2].

Joint. We present a joint view displaying the dependency trees for each sentence in parallel (see Figure 1). The visualization defaults to omitting inactive tokens, although they may be re-enabled through a checkbox at the top.

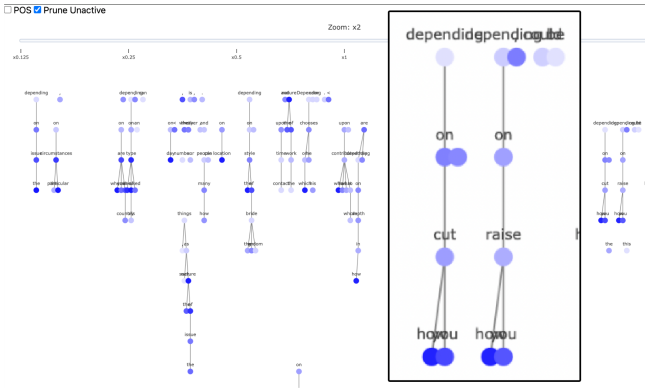

Figure 1: The joint view for this Gemma-9b Layer 11 feature visually demonstrates feature activations over *depending* and *on*, followed by a noun phrase. The following phrase (e.g. "how you") typically receives higher activations than the shared tokens, which is not obvious from text contexts.

Merged. We also introduce a second view to display merged sentences (see Figure 2). If a token sequence matched an existing branch, it is subsumed into the branch and visually emphasized. This view does not incorporate syntactic information. We pivoted to this view after receiving feedback that other syntactic structures were hard to understand.

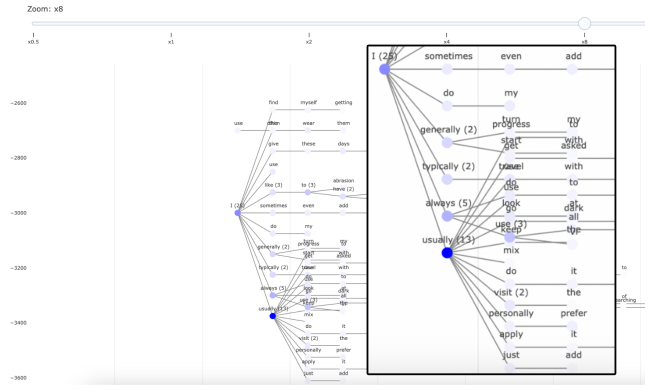

Figure 2: The initial token activations *I* and *usually* are shared across many contexts. The merged view for Gemma-9b Layer 11 feature displays a single tree representation of the shared tokens, and notes recurring sequences in the text ("I usually use").

4 Research Findings

We were able to identify several features through our visualization that could be easily identified based on shared activations. These features are not easily visualized using previously existing tools and primarily consisted of phrases with shared following contexts. **Depending on [P]** and **I usually [VP]** are two examples shown above;

[1] See [5] for details on the dependency parser used.

[2] The Gemma-9b features are available at the following dataset

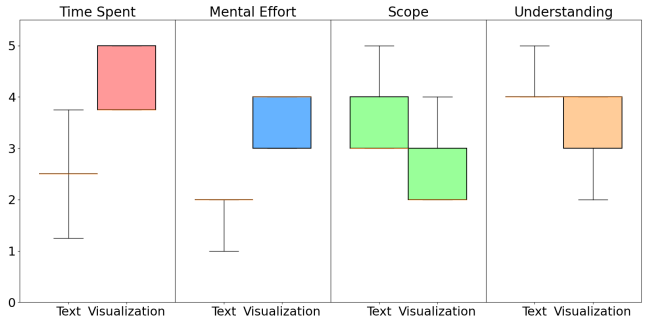

Figure 3: For the users in our online study (n=5), the merged view achieved comparable understanding of the feature and scope, but expended significantly more time and effort.

emphatic **do** phrases, **be clear**, and **too [adjP]** are other examples identified.

5 User Studies

We conducted two user studies involving an expert researcher and researchers from online interpretability communities.

We interviewed a researcher who specialized in SAE interpretability. They mentioned that existing metrics for dealing with high-dimensional space often did not characterize features well. They identified the merged view as the most promising and indicated that features in which contexts had longer syntactic regularities would be crucial in a proof-of-concept for a syntactic visualization.

The second study performed a head-to-head comparison of the text baseline and merged visualization. Users were shown a feature through both methods, and asked to describe what the pattern was across contexts. Users found the merged view less informative and intuitive than text contexts. Though these negative results could be partially attributed to overlapping text in the merged view, they additionally point towards the challenge of improving textual comprehension through visualization techniques.

6 Discussion

In this work, we have developed several methods for utilizing syntactic information inherent to a feature's activating contexts. We have found that UMAP plots of the feature space cluster based on part of speech distribution, and identified certain features which are well-characterized by the visualizations developed. The results of the user study then raise the question: **how might feature with highly regular syntactic activations be systematically determined?** Syntax-based feature identification would enable researchers to view a relevant subset of features, allowing them to compare and group them more easily.

Our user study tested one prototype and did not evaluate any syntactic views between feature contexts. Unfortunately, the primary negative feedback we received was about text overlapping and display, which precluded observations on the significance of the visualization overall. As seen in Figure 1, syntactic information is able to disambiguate regular from irregular feature activations. Further research might focus on how best to present syntactic regularities for visualization and comparison.

Acknowledgements

The authors wish to thank Professor David Laidlaw and their collaborator, Gonçalo Paulo, for their guidance and support throughout this project. The authors also thank their classmates for their feedback throughout the various stages of the research process.

REFERENCES

[1] P. L. J. I. B. L. S. N. N. Bart Bussmann, Michael Pearce. Showing sae latents are not atomic using meta-saes, 2024. Accessed: 2024-12-08.

[2] T. Bricken, A. Templeton, J. Batson, B. Chen, A. Jermyn, T. Conerly, N. Turner, C. Anil, C. Denison, A. Askell, R. Lasenby, Y. Wu, S. Kravec, N. Schiefer, T. Maxwell, N. Joseph, Z. Hatfield-Dodds, A. Tamkin, K. Nguyen, B. McLean, J. E. Burke, T. Hume, S. Carter, T. Henighan, and C. Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

[3] J. Dunefsky, P. Chlenski, and N. Nanda. Transcoders find interpretable llm feature circuits. *arXiv preprint arXiv:2406.11944*, 2024.

[4] W. Gurnee, T. Horsley, Z. C. Guo, T. R. Kheirkhah, Q. Sun, W. Hathaway, N. Nanda, and D. Bertsimas. Universal neurons in gpt2 language models. *arXiv preprint arXiv:2401.12181*, 2024.

[5] M. Honnibal and M. Johnson. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1373–1378, 2015.

[6] T. Lieberum, S. Rajamanoharan, A. Conmy, L. Smith, N. Sonnerat, V. Varma, J. Kramár, A. Dragan, R. Shah, and N. Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*, 2024.

[7] A. Makelov, G. Lange, and N. Nanda. Towards principled evaluations of sparse autoencoders for interpretability and control. *arXiv preprint arXiv:2405.08366*, 2024.

[8] G. Penedo, H. Kydlíček, A. Lozhkov, M. Mitchell, C. Raffel, L. Von Werra, T. Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*, 2024.

[9] S. Rajamanoharan, T. Lieberum, N. Sonnerat, A. Conmy, V. Varma, J. Kramár, and N. Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *arXiv preprint arXiv:2407.14435*, 2024.

[10] T. C. Team. Crosscoders: A transformer circuits analysis, 2024. Accessed: 2024-12-08.

[11] A. Templeton, T. Conerly, J. Marcus, J. Lindsey, T. Bricken, B. Chen, A. Pearce, C. Citro, E. Ameisen, A. Jones, H. Cunningham, N. L. Turner, C. McDougall, M. MacDiarmid, C. D. Freeman, T. R. Sumers, E. Rees, J. Batson, A. Jermyn, S. Carter, C. Olah, and T. Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024.

Eric Xia - Individual Contributions

Intellectual Contributions

As the primary investigator for the project, I drove ideation and prototyping of the visualization. I initiated contact with the collaborator, and wrote the initial proposal. I identified research questions and the initial motivations for the project. I summarized survey feedback and revised the user study according to the comments of classmates. I provided feedback and guidance to Byron on working with part of speech statistics. I was the primary contributor to many sections of the final report. Finally, I identified future research directions and open questions.

I also contributed to each of the weekly presentations and survey development. These included the initial in-class user study, the revised interpretability researcher study, the week 2 presentation examples, the week 3 presentation examples, the week 4 slides, the week 5 joint examples and study reflection, the week 6 results, and the final presentation.

Lastly, I was responsible for identifying and setting up meetings with interested parties, including extended conversations with Neuronpedia developers, and the user interview with Curt Tigges, Head of Science at Decode Research.

Technical Contributions.

I wrote the majority of the syntactic processing and token alignment code, which is present in the 'graphs' module referenced in the final code repository. These included the following technical contributions: Converting the raw token and location data into feature-specific dictionaries. Using the SpaCy dependency parser to convert tokens into tagged words, and adding token activation as an attribute. Using the SpaCy sentence tagger to extract relevant context. Writing tree merge functions, including identifying common nodes by lemma, counting total activations for each tree, and a subtree matching algorithm. Caching feature parses, contexts and activations to database to enable fast and frictionless retrieval. Finally, I implemented various text processing utilities, which converted between batch, sentence, and character indices.

I also created three main individual feature views with Plotly, Javascript, and the HTML/Jinja2 templating language, which are present in the 'templates' and 'static' folders in the final code repository. These front-end graph views started from LLM-generated base visualizations and were heavily modified. The views were served with Flask on a DigitalOcean Droplet virtual machine, and served as the focus for the user study. My technical contributions to the visualization included the following: introducing color gradients for activation values. Modifying a recursive node and edge display algorithm to prevent overlapping nodes. Adding the option to hide inactive nodes. Creating a custom text tag which updated dynamically with the node Part-of-Speech or text. Adding zoom functionality to the joint and merged views. Finally, I implemented the UMAP scatter navigation used in the class study.

Byron Butaney - Individual Contributions

Intellectual Contributions

My intellectual and practical contributions consisted of various milestones that changed throughout the course of the project. My intellectual contributions included meeting and ideating with Eric and our collaborator on a weekly basis, coming up with ways to create a UMAP that would provide more/different information compared to the Neuronpedia UMAP, and working with Eric to devise open research questions generated from our projects. I also helped devise, structure, and deploy the student and expert user studies. Finally, Eric and I met with Curt, the Neuronpedia developer, to demonstrate our tool, gain more insight into the problem space, and revise our user survey to be more suited to online expert users.

Technical Contributions

One of my overarching contributions was to identify any highly syntactic activating contexts for features. This included finding features that had significant connections between contexts and computing the part-of-speech distribution for every activating context of every feature. Another overarching goal was to apply dimensionality reduction techniques to the features in order to highlight any clustering by both their part of speech distribution as well as their most dominant part of speech. To visualize features by most dominant part of speech, I first computed the most-dominant part of speech for each feature based on the most frequently occurring POS context in that feature's list of contexts. I then labeled each feature and applied a UMAP with these labels included. To generate a visualization by part of speech distribution, I generated a list where each feature was given a row of 11 values (one for each possible POS). I then filled in these rows with the percentages that each POS appeared in the feature's activating contexts. UMAP-ing this data allowed us to see more clear clusters. This UMAP served as the basis for the UMAP used in our online tool. Since the data processing took such a long time, I ran the preprocessing locally on my computer over a couple of nights and saved the results as .npy files to speed up our visualization tool. I worked with Eric to design and conduct the three user studies, to create the final presentation, and to write the final report.